PingFederate®
Version 7.2

Administrator’s Manual
| Using Virtual Host Names | .................................................. | 73 |
| Changing Configuration Parameters | .................................................. | 74 |
| Installing a New License Key | .................................................. | 77 |
| Automating Configuration Migration | .................................................. | 78 |
| Copying the Key from the Source to the Target Server | .................................................. | 79 |
| Administrative Console Migration | .................................................. | 79 |
| Using the Migration Tool | .................................................. | 80 |
| Outbound Provisioning CLI | .................................................. | 83 |
| Customizing User-Facing Screens | .................................................. | 87 |
| IdP User-Facing Pages | .................................................. | 88 |
| SP User-Facing Pages | .................................................. | 90 |
| Either IdP or SP User-Facing Pages | .................................................. | 91 |
| OAuth User-Facing Pages | .................................................. | 92 |
| Localization | .................................................. | 92 |
| Overriding Locales with Cookies | .................................................. | 93 |
| Retrieving Localized Messages | .................................................. | 93 |
| Configuring Password Policy | .................................................. | 94 |
| Extending the Lifetime of the PF Cookie | .................................................. | 94 |
| Adding Custom HTTP Response Headers | .................................................. | 95 |

### Chapter 3 System Settings .................................................. 97

<p>| Chapter 3 | System Settings .................................................. 97 |
| Managing Server Settings | .................................................. | 97 |
| Setting Administration Options | .................................................. | 98 |
| Entering System Information | .................................................. | 99 |
| Configuring Runtime Notifications | .................................................. | 99 |
| Configuring Runtime Reporting | .................................................. | 100 |
| Using SNMP Monitoring | .................................................. | 101 |
| Runtime Monitoring Using JMX | .................................................. | 102 |
| Managing Accounts | .................................................. | 105 |
| Choosing Roles and Protocols | .................................................. | 106 |
| Specifying Federation Information | .................................................. | 109 |
| Setting System Options | .................................................. | 111 |
| Disabling Automatic Connection Validation | .................................................. | 111 |
| Defining Data-Store Validation Intervals | .................................................. | 112 |
| Defining proxy options | .................................................. | 112 |
| Managing System Updates | .................................................. | 113 |
| Configuring Outbound Provisioning Settings | .................................................. | 114 |
| Configuring Auto-Connect Metadata Signing | .................................................. | 115 |
| Configuring Auto-Connect Metadata Lifetime | .................................................. | 116 |
| Saving and Editing Server Settings | .................................................. | 117 |
| Managing Data Stores | .................................................. | 117 |
| Configuring a JDBC Database Connection | .................................................. | 119 |
| Setting Advanced Options | .................................................. | 121 |
| Configuring an LDAP Connection | .................................................. | 122 |
| Defining an LDAP Type | .................................................. | 124 |
| Setting Advanced LDAP Options | .................................................. | 125 |
| Specifying LDAP Binary Attributes | .................................................. | 127 |</p>
<table>
<thead>
<tr>
<th>Chapter 4</th>
<th>OAuth Configuration</th>
<th>155</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Enabling the OAuth AS</td>
<td>155</td>
</tr>
<tr>
<td></td>
<td>Using OAuth Menu Selections</td>
<td>156</td>
</tr>
<tr>
<td></td>
<td>Centralized Session Management</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>Asynchronous Front-Channel Logout</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>Back-Channel Session Revocation</td>
<td>158</td>
</tr>
<tr>
<td></td>
<td>Authorization Server Settings</td>
<td>159</td>
</tr>
<tr>
<td></td>
<td>Access Token Management</td>
<td>165</td>
</tr>
<tr>
<td></td>
<td>Configuring Reference-Token Management</td>
<td>165</td>
</tr>
<tr>
<td></td>
<td>Configuring JSON-Token Management</td>
<td>167</td>
</tr>
<tr>
<td></td>
<td>Defining the Access Token Attribute Contract</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td>Using the Access Token Management Summary Screen</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td>Configuring OpenID Connect Policies</td>
<td>170</td>
</tr>
<tr>
<td></td>
<td>Configuring the Policy Attribute Contract</td>
<td>171</td>
</tr>
<tr>
<td></td>
<td>Policy Attribute Sources and User Lookup</td>
<td>172</td>
</tr>
<tr>
<td>Contents</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>Policy Contract Fulfillment</td>
<td>173</td>
<td></td>
</tr>
<tr>
<td>Identifying Issuance Criteria for Policy Mapping</td>
<td>174</td>
<td></td>
</tr>
<tr>
<td>Using the Policy Summary Screen</td>
<td>176</td>
<td></td>
</tr>
<tr>
<td>Client Management</td>
<td>177</td>
<td></td>
</tr>
<tr>
<td>Configuring a Client</td>
<td>178</td>
<td></td>
</tr>
<tr>
<td>Resource-Owner Credentials Mapping</td>
<td>181</td>
<td></td>
</tr>
<tr>
<td>Resource-Owner Contract Fulfillment</td>
<td>182</td>
<td></td>
</tr>
<tr>
<td>Identifying Issuance Criteria for Credentials Mapping</td>
<td>183</td>
<td></td>
</tr>
<tr>
<td>Using the Credentials-Mapping Summary Screen</td>
<td>185</td>
<td></td>
</tr>
<tr>
<td>IdP Adapter Mapping for OAuth</td>
<td>185</td>
<td></td>
</tr>
<tr>
<td>Grant Contract Fulfillment</td>
<td>186</td>
<td></td>
</tr>
<tr>
<td>Defining Issuance Criteria for OAuth Adapter Mapping</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>Using the Adapter-Mapping Summary Screen</td>
<td>189</td>
<td></td>
</tr>
<tr>
<td>Access Token Mapping</td>
<td>189</td>
<td></td>
</tr>
<tr>
<td>Token Attribute Contract Fulfillment</td>
<td>190</td>
<td></td>
</tr>
<tr>
<td>Configuring Issuance Criteria for Access Token Mapping</td>
<td>191</td>
<td></td>
</tr>
<tr>
<td>Using the Token-Mapping Summary Screen</td>
<td>193</td>
<td></td>
</tr>
<tr>
<td>Configuring an OAuth SAML Grant IdP Connection</td>
<td>193</td>
<td></td>
</tr>
<tr>
<td>Specifying an Attribute Contract for the OAuth SAML Grant</td>
<td>194</td>
<td></td>
</tr>
<tr>
<td>Choosing a Data Store for OAuth SAML Grant Attribute Mapping</td>
<td>195</td>
<td></td>
</tr>
<tr>
<td>OAuth SAML Grant Contract Fulfillment</td>
<td>195</td>
<td></td>
</tr>
<tr>
<td>Selecting Issuance Criteria for OAuth SAML Grant</td>
<td>197</td>
<td></td>
</tr>
<tr>
<td>OAuth SAML Grant Configuration Summary</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>OAuth Attribute Mapping Using a Data Store</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>Defining a JDBC Location for OAuth</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>Configuring an OAuth Database Filter (WHERE Clause)</td>
<td>201</td>
<td></td>
</tr>
<tr>
<td>Searching LDAP for OAuth Mapping</td>
<td>201</td>
<td></td>
</tr>
<tr>
<td>Configuring an LDAP Filter for OAuth Mapping</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td>Configuring OAuth Custom Source Filters</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td>Selecting OAuth Custom Source Fields</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td>Chapter 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security Management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Certificate Management</td>
<td>203</td>
<td></td>
</tr>
<tr>
<td>Trusted Certificate Authorities</td>
<td>204</td>
<td></td>
</tr>
<tr>
<td>SSL Server Certificates</td>
<td>204</td>
<td></td>
</tr>
<tr>
<td>SSL Client Keys and Certificates</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>Digital Signing and Decryption Keys and Certificates</td>
<td>209</td>
<td></td>
</tr>
<tr>
<td>Certificate Revocation Checking</td>
<td>212</td>
<td></td>
</tr>
<tr>
<td>Authentication</td>
<td>215</td>
<td></td>
</tr>
<tr>
<td>Application Authentication</td>
<td>216</td>
<td></td>
</tr>
<tr>
<td>Validating Password Credentials</td>
<td>217</td>
<td></td>
</tr>
<tr>
<td>Choosing a Type</td>
<td>218</td>
<td></td>
</tr>
<tr>
<td>Configuring the LDAP Credential Validator</td>
<td>219</td>
<td></td>
</tr>
<tr>
<td>Configuring the Simple Credential Validator</td>
<td>221</td>
<td></td>
</tr>
<tr>
<td>Configuring the RADIUS Credential Validator</td>
<td>222</td>
<td></td>
</tr>
<tr>
<td>Finishing the Validator Configuration</td>
<td>225</td>
<td></td>
</tr>
</tbody>
</table>
# Identity Provider SSO Configuration

## Application Integration Settings

- **Using AD Domains and Kerberos Realms** .................................................. 225
- **Configuring a Domain or Realm** ......................................................... 225
- **Managing Domain or Realm Settings** .................................................. 227

## Identity Provider Adapters

- **Configuring Adapters** ................................................................. 230
- **Selecting an IdP Adapter Type** ...................................................... 231
- **Configuring an IdP Adapter** .......................................................... 232
- **Invoking Adapter Actions** .............................................................. 233
- **Extending an Adapter Contract** ..................................................... 233
- **Setting Pseudonym Values and Masking** ........................................ 234
- **Selecting an Authentication Context** ............................................. 234
- **Editing and Saving Adapter Instances** ........................................... 235

## Configuring Adapter Selectors

- **Configuring Adapter Selectors** ...................................................... 235
- **Choosing a Selector Type** ............................................................ 236
- **Configuring the CIDR Adapter Selector** ........................................ 237
- **Configuring the Cluster Node Selector** ........................................... 237
- **Defining Cluster Node Results** ...................................................... 237
- **Configuring the Connection Set Selector** ....................................... 238
- **Configuring the HTTP Header Selector** .......................................... 238
- **Configuring the OAuth Scope Selector** ......................................... 239
- **Configuring the Requested AuthN Context Selector** ....................... 240
- **Defining AuthN Context Results** ................................................... 240
- **Finishing the Selector-Instance Configuration** ................................ 241
- **Mapping Selector Results to Adapter Instances** ............................. 242
- **Configuring a Default URL and Error Message** ............................. 243
- **Viewing Application Endpoints** .................................................... 244
- **Viewing Protocol Endpoints** .......................................................... 245

## Managing SP Connections

- **Managing SP Connections** .............................................................. 246
- **Accessing Connections** ................................................................. 247
- **Via the Main Menu** ................................................................. 247
- **Via the Manage Connections Screen** .......................................... 249
- **Choosing a Connection Template** ................................................. 252
- **Choosing a Connection Type** ....................................................... 252
- **Choosing Connection Options** ...................................................... 254
- **Importing Metadata** ................................................................. 254
- **Importing a Verification Certificate** ............................................. 255
- **Viewing the Metadata Summary** .................................................. 255

## General Information

- **General Information** ................................................................. 255
- **Configuring Browser-Based SSO** .................................................. 257
- **Configuration Steps** ................................................................. 257
- **Choosing Profiles (SAML 2.0)** ..................................................... 259
- **Setting an Assertion Lifetime** ...................................................... 261
- **Assertion Creation** ................................................................. 261
- **Configuring Protocol Settings** ..................................................... 290
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring Identity Store Provisioners</td>
<td>344</td>
</tr>
<tr>
<td>Creating an Identity Store Provisioner Instance</td>
<td>344</td>
</tr>
<tr>
<td>Defining Identity Store Provisioner Behavior</td>
<td>345</td>
</tr>
<tr>
<td>Extending Identity Store Provisioner Contracts</td>
<td>345</td>
</tr>
<tr>
<td>Editing and Saving Identity Store Provisioner Instances</td>
<td>346</td>
</tr>
<tr>
<td>Configuring Default URLs</td>
<td>346</td>
</tr>
<tr>
<td>Setting Target Validation</td>
<td>347</td>
</tr>
<tr>
<td>Viewing SP Application Endpoints</td>
<td>349</td>
</tr>
<tr>
<td>Federation Settings</td>
<td>350</td>
</tr>
<tr>
<td>Attribute Requester Mapping</td>
<td>350</td>
</tr>
<tr>
<td>Managing IdP Connections</td>
<td>353</td>
</tr>
<tr>
<td>Accessing IdP Connections</td>
<td>354</td>
</tr>
<tr>
<td>From the Main Menu</td>
<td>354</td>
</tr>
<tr>
<td>From the Manage Connections Screen</td>
<td>355</td>
</tr>
<tr>
<td>Choosing an IdP Connection Type</td>
<td>358</td>
</tr>
<tr>
<td>Choosing IdP Connection Options</td>
<td>359</td>
</tr>
<tr>
<td>Importing IdP Metadata</td>
<td>360</td>
</tr>
<tr>
<td>Importing a Certificate</td>
<td>360</td>
</tr>
<tr>
<td>Viewing the Summary</td>
<td>360</td>
</tr>
<tr>
<td>General Connection Information</td>
<td>360</td>
</tr>
<tr>
<td>Configuring Browser SSO</td>
<td>362</td>
</tr>
<tr>
<td>Connection Configuration Steps</td>
<td>363</td>
</tr>
<tr>
<td>Choosing SAML Profiles</td>
<td>364</td>
</tr>
<tr>
<td>User-Session Creation</td>
<td>367</td>
</tr>
<tr>
<td>Configuring OAuth Attribute Mapping</td>
<td>388</td>
</tr>
<tr>
<td>Configuring SAML Protocol Settings</td>
<td>392</td>
</tr>
<tr>
<td>Editing and Saving SSO Settings</td>
<td>400</td>
</tr>
<tr>
<td>Configuring the Attribute Query Option</td>
<td>401</td>
</tr>
<tr>
<td>Setting the Attribute Authority Service URL</td>
<td>401</td>
</tr>
<tr>
<td>Mapping Attribute Names</td>
<td>401</td>
</tr>
<tr>
<td>Defining Security Policy</td>
<td>402</td>
</tr>
<tr>
<td>Saving the Attribute Query Configuration</td>
<td>402</td>
</tr>
<tr>
<td>Using Just-in-Time Provisioning</td>
<td>403</td>
</tr>
<tr>
<td>Selecting Attribute Sources (SAML 2.0)</td>
<td>403</td>
</tr>
<tr>
<td>Identifying the User Repository</td>
<td>404</td>
</tr>
<tr>
<td>Specifying an LDAP User-Record Location</td>
<td>405</td>
</tr>
<tr>
<td>Defining an LDAP Filter</td>
<td>405</td>
</tr>
<tr>
<td>Identifying Provisioning Attributes for LDAP</td>
<td>406</td>
</tr>
<tr>
<td>Selecting a SQL Method</td>
<td>407</td>
</tr>
<tr>
<td>Specifying a Database User-Record Location</td>
<td>408</td>
</tr>
</tbody>
</table>
Specifying a Unique-ID Database Column ........................................ 409
Specifying a Stored-Procedure Location ........................................ 409
Mapping Attributes to a User Account ........................................... 410
Choosing an Event Trigger ........................................................... 412
Error Handling ............................................................................. 413
Using the Provisioning Summary Screen ........................................ 413
Configuring Inbound Provisioning .................................................. 413
Specifying the User Repository ..................................................... 414
Identifying an LDAP User-Record Location .................................... 415
Defining a Unique ID ..................................................................... 415
Writing User Information to the Data Store .................................... 417
Configuring a SCIM Response ......................................................... 420
Handling SCIM Delete Requests .................................................... 423
Saving the Inbound Provisioning Configuration ............................... 424
Configuring Security Credentials .................................................. 424
Back-Channel Authentication ......................................................... 424
Digital Signature Settings ............................................................... 427
Signature Verification Settings ....................................................... 428
Choosing an Encryption Certificate ............................................... 431
Choosing a Decryption Key ............................................................ 432
Saving Credential Configurations .................................................. 432
IdP Connection Activation and Summary ....................................... 433
Configuring IdP Auto-Connect ....................................................... 433
Configuring the Initial Setup .......................................................... 434
Choosing a Certificate ................................................................... 434
Configuring User-Session Creation ............................................... 435
Connection Activation and Summary .............................................. 435
Specifying Allowed IdP Domains ................................................... 436

Chapter 8  WS-Trust STS Configuration ........................................... 437
Server Settings .............................................................................. 437
Enabling the WS-Trust STS ........................................................... 437
Configuring STS Authentication ..................................................... 438
Selecting Authentication Methods .................................................. 439
Configuring Basic Authentication .................................................. 439
Configuring Mutual SSL Authentication ....................................... 441
Using the STS Summary Screen ..................................................... 443
IdP Configuration for STS ............................................................... 443
Configuring Token Processors ....................................................... 443
Selecting a Token Processor Type .................................................. 445
Configuring a SAML Token Processor Instance ............................. 445
Configuring an OAuth Token Processor Instance ........................... 447
Configuring a JSON Web Token (JWT) Processor Instance .......... 447
Configuring a Username Token Processor Instance ..................... 448
Extending a Processor Contract ..................................................... 449
Setting Attribute Masking ............................................................. 449
Appendix A OpenToken Adapter Configuration ................................................. 499
  Configuring the IdP OpenToken Adapter ................................................. 500
  Configuring the SP OpenToken Adapter .................................................. 504
Appendix B HTTP Basic Adapter Configuration ............................................. 509
  Configuring the HTTP Basic IdP Adapter .................................................. 509
Appendix C HTML Form Adapter Configuration ............................................. 515
  Configuring the HTML Form IdP Adapter .................................................. 515
Appendix D Composite Adapter Configuration .............................................. 523
  Configuring the Composite Adapter ....................................................... 523
Appendix E Application Endpoints ............................................................... 529
  IdP Endpoints ....................................................................................... 529
  SP Endpoints ....................................................................................... 533
    SP Services ....................................................................................... 533
    SCIM Inbound Provisioning Endpoints .................................................. 538
  System-Services Endpoints ..................................................................... 540
Appendix F OAuth 2.0 Endpoints ................................................................. 543
  Token Endpoint ..................................................................................... 543
  Client Identification and Authentication .................................................. 543
    Grant Type Parameters ........................................................................ 544
      Authorization Code Grant Type ......................................................... 544
      Refresh Token Grant Type .................................................................. 545
      Resource Owner Credentials (Password) Grant Type ......................... 545
      Client Credentials Grant Type ............................................................ 546
      SAML 2.0 Bearer Assertion Grant Type .............................................. 546
      Access Token Verification/Validation Grant Type ............................... 547
  Token Revocation Endpoint ..................................................................... 547
  Client Identification and Authentication for Token Revocation ................. 548
  Parameters ............................................................................................ 548
  Authorization Endpoint .......................................................................... 549
<table>
<thead>
<tr>
<th>Contents</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Grant-Management Endpoint</td>
<td>551</td>
</tr>
<tr>
<td>OpenID Connect Metadata Endpoint</td>
<td>551</td>
</tr>
<tr>
<td><strong>Appendix G</strong></td>
<td>553</td>
</tr>
<tr>
<td>Web Service Interfaces</td>
<td></td>
</tr>
<tr>
<td>Connection Management Service</td>
<td>554</td>
</tr>
<tr>
<td>Exporting a Connection</td>
<td>554</td>
</tr>
<tr>
<td>Exporting Manually</td>
<td>554</td>
</tr>
<tr>
<td>Using the Connection Service</td>
<td>555</td>
</tr>
<tr>
<td>Code Sample</td>
<td>555</td>
</tr>
<tr>
<td>Importing Connections</td>
<td>555</td>
</tr>
<tr>
<td>Sample Code</td>
<td>556</td>
</tr>
<tr>
<td>Deleting Connections</td>
<td>556</td>
</tr>
<tr>
<td>Code Example</td>
<td>556</td>
</tr>
<tr>
<td>Cluster Configuration Replication</td>
<td>556</td>
</tr>
<tr>
<td>Example Code</td>
<td>557</td>
</tr>
<tr>
<td>Validation Disclaimer</td>
<td>557</td>
</tr>
<tr>
<td>SSO Directory Service</td>
<td>557</td>
</tr>
<tr>
<td>Coding Example</td>
<td>558</td>
</tr>
<tr>
<td>SOAP Request and Response Example</td>
<td>559</td>
</tr>
<tr>
<td>OAuth Client Management Service</td>
<td>560</td>
</tr>
<tr>
<td>OAuth Access Grant Management Service</td>
<td>566</td>
</tr>
<tr>
<td>Session Revocation API</td>
<td>568</td>
</tr>
<tr>
<td>PingFederate Administrative API</td>
<td>570</td>
</tr>
<tr>
<td>Using the API Interactive Documentation</td>
<td>570</td>
</tr>
<tr>
<td><strong>Appendix H</strong></td>
<td>573</td>
</tr>
<tr>
<td>Using Attribute Mapping Expressions</td>
<td></td>
</tr>
<tr>
<td>Enabling and Disabling Expressions</td>
<td>573</td>
</tr>
<tr>
<td>Constructing Expressions</td>
<td>574</td>
</tr>
<tr>
<td>Data Store Syntax</td>
<td>575</td>
</tr>
<tr>
<td>Issuance Criteria Syntax</td>
<td>575</td>
</tr>
<tr>
<td>Expression Examples</td>
<td>576</td>
</tr>
<tr>
<td>Issuance Criteria and Multiple Virtual Server IDs</td>
<td>576</td>
</tr>
<tr>
<td>Using the OGNL Edit Screen</td>
<td>578</td>
</tr>
<tr>
<td><strong>Appendix I</strong></td>
<td>581</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td></td>
</tr>
<tr>
<td>Data Store Issues</td>
<td>581</td>
</tr>
<tr>
<td>Installation Issues</td>
<td>582</td>
</tr>
<tr>
<td>Runtime Issues</td>
<td>582</td>
</tr>
<tr>
<td>Server Startup</td>
<td>582</td>
</tr>
<tr>
<td>Glossary</td>
<td>583</td>
</tr>
<tr>
<td>List of Acronyms</td>
<td>589</td>
</tr>
</tbody>
</table>
Preface

About This Manual

This manual provides information about using Ping Identity’s PingFederate to deploy a secure single sign-on (SSO) solution based on the latest security and e-business standards.

Overview

The manual consists of:

- **Chapter 1, “Key Concepts”** — A discussion of central concepts needed to understand SSO, the WS-Trust Security Token Service (STS), and PingFederate deployment and administration.
- **Chapter 2, “System Administration”** — Information about maintaining the PingFederate server and deployment, using log files, managing users, and handling other administrative functions.
- **Chapter 3, “System Settings”** — How to configure your local PingFederate server settings.
- **Chapter 4, “OAuth Configuration”** — How to configure PingFederate to act as an OAuth Authorization Server.
- **Chapter 5, “Security Management”** — Information about importing, exporting, and maintaining certificates and keys in PingFederate.
- **Chapter 6, “Identity Provider SSO Configuration”** — How to configure PingFederate to act as an Identity Provider (IdP) and establish connections to Service Providers.
- **Chapter 7, “Service Provider SSO Configuration”** — How to configure PingFederate to act as a Service Provider (SP) and establish connections to Identity Providers.
- **Chapter 8, “WS-Trust STS Configuration”** — How to configure PingFederate to act as a Security Token Service for Web Service Clients and Providers in either an IdP or SP environment.
Preface

- Appendix A, “OpenToken Adapter Configuration” — How to configure PingFederate to use the packaged OpenToken Adapter for interfacing with your Web applications.
- Appendix B, “HTTP Basic Adapter Configuration” — How to configure PingFederate to use the packaged HTTP Basic Adapter.
- Appendix C, “HTML Form Adapter Configuration” — How to configure PingFederate to use the packaged HTML Form Adapter.
- Appendix D, “Composite Adapter Configuration” — How to configure PingFederate to use the packaged Composite Adapter.
- Appendix E, “Application Endpoints” — Detailed information about using PingFederate connection endpoints for Web single sign-on and single logout.
- Appendix G, “Web Service Interfaces” — Information administrators and developers can use to automate connection-configuration management, runtime SSO partner discovery, and OAuth client configurations.
- Appendix H, “Using Attribute Mapping Expressions” — How to enable and use expressions in conjunction with mapping attributes.
- Appendix I, “Troubleshooting” — Solutions for difficulties that may be encountered.
- Glossary — Definitions of terms used in the manual and in identity federation parlance.
- List of Acronyms

Intended Audience

This manual is intended for security and network administrators and other IT professionals responsible for identity management among business entities, both internal and external.

**Note:** The information in this manual is presented from the viewpoint of an administrative user with full permissions (see “Account Management” on page 64).

Text Conventions

This document uses the text conventions identified below.

Table 1: Text Conventions

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Width</td>
<td>Indicates text that must be typed exactly as shown in the instructions. Also used to represent program code, file names, and directory paths.</td>
</tr>
<tr>
<td>Blue text</td>
<td>Indicates hypertext links.</td>
</tr>
<tr>
<td><em>Italic</em></td>
<td>Used for emphasis and document titles.</td>
</tr>
<tr>
<td>▶️ [text]</td>
<td>Used for procedures where only one step is required.</td>
</tr>
</tbody>
</table>
Table 1: Text Conventions (Continued)

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sans serif</td>
<td>Identifies descriptive text on a user-interface screen. Example: “Print Document dialog”</td>
</tr>
<tr>
<td>Sans serif bold</td>
<td>Identifies menu items, navigational links, or buttons. For example: Click Save.</td>
</tr>
</tbody>
</table>

Other Documentation

The documents listed below are available under Product Documentation at pingidentity.com.

Tip: PingFederate provides context-sensitive Help. Click Help in the upper-right portion of the administrative console for immediate, relevant guidance and links to related information.

Getting Started – Provides an introduction to secure Internet SSO and PingFederate, including background information about federated identity management and standards, product installation instructions, and a primer on using the PingFederate administrative console.

Integration Overview – A high-level description of options available for integrating identity-management systems and applications with PingFederate.

Server Clustering Guide – Describes how to deploy PingFederate in a cluster to increase throughput and availability.

SDK Developer’s Guide – Provides technical guidance for using the Java Software Developer Kit for PingFederate.

Quick-Start Guide – Provides instructions for deploying a preconfigured PingFederate server to run with demonstration Web applications. Newcomers to PingFederate may wish to follow this Guide as a first step to establishing a simple SSO identity federation between two Web applications and to become familiar with PingFederate. The Guide is contained in a separate quick-start package available for download on the Ping Identity Web site.

Web Resources – Ping Identity continually updates its Resource Center (www.pingidentity.com/resource-center) with general and technical information in the form of white papers, demonstrations, webinars, and other resources.

Note: If you encounter any difficulties with configuration or deployment, please look for help at the Ping Identity Support Center (www.pingidentity.com/support).

PingFederate documents may include hypertext links to third-party Web sites that provide installation instructions, file downloads, and reference documentation. These links were tested prior to publication, but they may not remain current throughout the life of these documents. Please contact Ping Identity Support (www.pingidentity.com/support) if you encounter a problem.
This chapter provides background information and preparation to help administrators understand and use PingFederate.

- “Connection Types” below
- “About WS-Trust STS” on page 6
- “About OAuth” on page 10
- “SSO Integration Kits and Adapters” on page 14
- “Hierarchical Plug-in Configurations” on page 17
- “Identity Mapping” on page 17
- “About Attributes” on page 19
- “Security Infrastructure” on page 27
- “Using Auto-Connect” on page 31
- “User Provisioning” on page 35
- “Federation Planning Checklist” on page 37

**Tip:** For an introduction to secure single sign-on (SSO), federated identity management, and PingFederate product features, see the “About Identity Federation and SSO” chapter in Getting Started.

### Connection Types

PingFederate features an integrated administrative console for configuring four kinds of connections to identity-federation partners:

- **Browser-based SSO** – Also called Browser SSO in the administrative console, this term is often used to refer to standards-based secure SSO, which
generally depends on a user’s browser to transport identity assertions and other messaging between partner endpoints (see the “Supported Standards” chapter in Getting Started).

- WS-Trust STS – Employs the PingFederate Security Token Service (STS), which enables Web Service Client and Provider applications to extend SSO to identity-enabled Web Services at provider sites, using another set of standards (see the next section, “About WS-Trust STS”). These standards, including WS-Trust, do not rely on the user's browser for message transport.

- OAuth SAML Grant – Exchanges a SAML assertion for an OAuth access token with the PingFederate Authorization Server (AS) (see “About OAuth” on page 10).

- Provisioning – Provides automated cross-domain inbound and outbound user management (see “User Provisioning” on page 35).

The types of connections can be configured together for the same partner or independently.

**About WS-Trust STS**

The PingFederate WS-Trust STS allows organizations to extend SSO identity management to Web Services. (For information about WS-Trust and the role of an STS, see “Web Services Standards” in the “Supported Standards” chapter of Getting Started.)

The WS-Trust STS can be configured for partner connections independently or in conjunction with browser-based SSO for either an IdP or an SP deployment. The STS is bundled with separate plug-ins for standard SAML (Security Assertion Markup Language) token processing and generation (see “Token Processors and Generators” on page 7).

**Connection-Based Policy**

For both the IdP and SP roles, PingFederate employs a partner-connection configuration, which enables the association of Web Services authentication policies with federation partners. For STS processing, these policies define configurations for handling WS-Trust requests and transferring identity information between security domains (see “Web Services Standards” in the “Supported Standards” chapter of Getting Started).

**IdP Configuration**

In an IdP role, you use the administrative console to configure WS-Trust request-processing policy for your SP partner including:

- The type of SAML token to create—suitable for consumption by the intended Web Service Provider (WSP, at the SP site)—in response to an “Issue” request from a Web Service Client (WSC) application

- The mapping of attributes to include within the issued SAML token

- The key used to create a digital signature for the issued SAML token
SP Configuration

In an SP role, you use the administrative console to configure WS-Trust request-processing policy for your IdP partner including:

- Whether to validate the incoming SAML token only, or to validate the incoming token and also issue a local token
- The mapping of attributes to include in the locally issued token (when applicable)
- The certificate used to verify the digital signature for the incoming SAML token
- The key used to decrypt the incoming SAML token (when needed)

Token Processors and Generators

PingFederate provides support for a variety of security-token formats, through token processors and generators that plug into the PingFederate server. These plug-ins deploy similarly to browser-based SSO adapters (see “SSO Integration Kits and Adapters” on page 14).

For an IdP, token processors provide a mechanism through which PingFederate can validate an incoming token from a WSC and map attributes to be included in the issued SAML token.

For an SP, token generators provide a mechanism through which PingFederate can generate a local token based upon the incoming SAML token from a WSP and map attributes to be included in that token.

Only SAML 1.1 or 2.0 tokens are generated by PingFederate configured as an IdP for sending across trust boundaries to a federated SP partner. Likewise, only SAML tokens are accepted by PingFederate configured as an SP. Token plug-ins allow a modular approach for validating and producing the various token types used by different applications or systems within a conceptual trust domain.

PingFederate provides bundled and separately available token plug-ins.

Tip: For direct STS token exchange within the same domain or trust boundary, you can also use the PingFederate STS to exchange one token type for another directly, without generating a transitional SAML token (see “STS Token Exchange Mapping” on page 146).

PingFederate also allows you to use a configuration of a token processor or generator as a parent instance from which you can create child instances (see “Hierarchical Plug-in Configurations” on page 17).

Bundled Token Plug-ins

PingFederate is installed with token processors for an IdP configuration that accept and validate SAML 1.1, 2.0 tokens, OAuth Bearer access tokens, and JWT tokens (see “Token Models” on page 10). (SAML tokens are issued on the IdP side via built-in browser-based SSO capabilities.)

For an SP configuration, token generators are provided for issuing local SAML 1.1 or 2.0 tokens. (Incoming SAML tokens are validated, once again, by using built-in capabilities.)
Chapter 1  
Key Concepts

**Commercial Token Plug-ins**

Ping Identity provides token plug-ins to work with various authentication systems and leading identity management systems. Available plug-ins, together known as **Token Translators**, may be downloaded from the Ping Identity Web site (www.pingidentity.com/support-and-downloads).

**WSC and WSP Support**

Ping Identity provides the Java client Software Development Kit (SDK) for enabling Web Service applications (WS clients or providers) to interact with the PingFederate STS.

In addition, for WSC STS clients PingFederate provides built-in protocol support for Windows Identity Foundation (WIF) applications based on the Windows Communication Foundation (WCF) framework.

*Note:* The WIF framework includes WS-* protocol support and can interact natively with PingFederate.

**Client SDK**

The STS Java client SDK provides interfaces that create the WS-Trust Request Security Token (RST) and Request Security Token Response (RSTR) messaging to interact with the PingFederate STS endpoints. Using the SDK library, applications are not responsible for forming these WS-Trust protocol messages, and instead interact only with the tokens themselves.

The SDK is available for download at the Ping Identity Web site (www.pingidentity.com/support-and-downloads).

**Windows Identity Foundation Clients**

PingFederate natively supports STS clients using claims-based WIF technology. Claims-based federated identity for Web Services is a part of the WS-Trust standard that permits client applications to make access-policy decisions, when specifically categorized user attributes are sent in the security token (see “STS Namespaces” on page 22).

The PingFederate STS supports the following bindings in the .NET federated-security scenarios with WS-Trust:

- WSFederationHttpBinding
- WS2007FederationHttpBinding

Additionally, the PingFederate STS supports the following bindings for RST and RSTR interactions with .NET. (Support for these bindings is limited to the Username, x509, SAML 1.1, and SAML 2.0 token types.)

- WSHttpBinding
- WS2007HttpBinding

*Note:* For token types such as Kerberos, where customizing default bindings may be necessary, the PingFederate STS supports the use of customBinding.
For more information about bindings, see Microsoft's System-Provided Bindings (msdn.microsoft.com/en-us/library/ms730879.aspx).

Developers can obtain metadata from PingFederate to expedite configuring their applications. PingFederate offers two varieties of metadata, which are often used together to arrive at functional WSC and WSP configurations:

- Federation Metadata, which contains details on the PingFederate public signing certificate and other information required to establish the trust relationship (see “/pf/federation_metadata.ping” on page 541).
- Metadata Exchange, which contains connection details relating to the SP partner (see “/pf/sts_mex.ping” on page 541).


**STS OAuth Integration**

PingFederate STS provides several ways to facilitate the use of issued tokens with an OAuth Authorization Server (AS) (see the next section, “About OAuth” on page 10).

**OAuth Token Processor**

This token processor provides a mechanism through which PingFederate STS can validate an incoming OAuth Bearer access token. The token processor reads and validates the access token and returns any additional user attributes defined (see “Configuring an OAuth Token Processor Instance” on page 447 and “Defining the Access Token Attribute Contract” on page 170).

**SAML 2.0 Bearer Assertion Grant Type**

urn:ietf:params:oauth:grant-type:saml2-bearer

This token request returns an encoded SAML assertion that a WSC can use to request OAuth access tokens from any OAuth AS that supports the SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization specification (http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer). This capability provides a bridge between the WS-Trust client-STS relationship and the trust relationship the same client may have with an OAuth AS, allowing the client to obtain additional resources on behalf of already-authenticated users in follow-on transactions.

**OAuth Access Token via SAML 2.0 Bearer Assertion**


This proprietary token request is similar to the one above but returns an OAuth access token directly. Acting as an IdP, PingFederate generates the intermediate, encoded SAML assertion and requests an access token from the OAuth AS on behalf of the WSC. (The AS endpoint is obtained from the element <AppliesTo> of the WS-Trust RST message.)
Chapter 1
Key Concepts

About OAuth

PingFederate can be configured to act as an OAuth Authorization Server (OAuth AS), allowing a resource owner (typically, an end user) to grant authorization to an OAuth Client requesting access to resources hosted by a Resource Server (RS). (For information about OAuth and the role of an AS, see “OAuth 2.0” in the “Supported Standards” chapter of Getting Started.) The OAuth AS issues tokens to clients on behalf of a resource owner for use in authenticating a subsequent API call to the RS—typically, but not exclusively, a Representational State Transfer (REST) API call.

Note: OAuth AS capabilities are available under special licensing. If your license does not include the OAuth AS, please contact sales@pingidentity.com.

The PingFederate OAuth AS can be configured independently or in conjunction with STS or browser-based SSO for either an IdP or an SP deployment. In an SP deployment, the inbound SAML assertion can be used to provide authentication information about the user that can be associated with the access token (see “Configuring OAuth Attribute Mapping” on page 388). In an IdP deployment, an IdP adapter can be used to authenticate and provide user information for the access token. See “OAuth Configuration” on page 155 for more information.

For an STS IdP, an OAuth token processor is provided with the PingFederate installation (see “IdP Configuration for STS” on page 443).

Delegated Access Types

Explicit Delegation – This is the most common OAuth use case, which involves a resource owner who explicitly delegates to a client the authority to make API calls to a Resource Server and is asked to approve the transaction. This is the type of delegation inherent in Web Redirect transactions (see “Web Redirect Flow” in the “Supported Standards” chapter of Getting Started).

Implicit Delegation – Implicit delegation also generally involves a client who calls an API on behalf of a user; however, the client’s authority is implied by the nature of the transaction, and the user is not specifically asked to approve the transaction.

Token Models

Successful OAuth transactions require an OAuth AS to issue access tokens for use in authenticating an API call. These tokens may be characterized by both their data model and security model.

Note: PingFederate supports different token models through a plug-in architecture (see “Access Token Management” on page 165).

Token Data Model

A token data model refers to whether the token carries identity and security information or acts as a pointer to the information.
Self-Contained Tokens – Contain identity and security information and attributes in a transport format such as JSON (JavaScript Object Notation), signed by the AS and verified directly by the RS.

Reference Tokens – Serve as a reference to some set of attributes. The RS must de-reference the token for the corresponding identity and security information at the OAuth AS that issued it.

Token Security Model

A token security model refers to the conditions that must be met by a client in order to use a token on an API call. The currently supported model is a Bearer Token—a client's presentation of the token to the RS (for example, as a parameter on the API call) is interpreted as providing sufficient proof to the RS that the client received the same token from the OAuth AS.

Grant Types

To obtain an access token, a client interacts with an OAuth AS, sending a request for an access token that includes an access grant. An access grant is also used when an RS requests validation of an access token from the OAuth AS.

Primary Grant Types

OAuth defines several different access grant types—each reflecting different authorization mechanisms.

Authorization Code – An authorization code is returned to the client through a browser redirect after the resource owner gives consent to the OAuth AS. The client subsequently exchanges the authorization code for an access (and often a refresh) token. Resource owner credentials are never exposed to the client.

Implicit – An access token is returned to the client through a browser redirect in response to the resource owner authorization request (rather than an intermediate authorization code). This grant type is suitable for clients incapable of keeping client credentials confidential (for use in authenticating with the OAuth AS) such as client applications implemented in a browser using a scripting language like Javascript.

Resource Owner Credentials – The client collects the resource owner's password and exchanges it at the OAuth AS for an access token, and often a refresh token (see below). This grant type is suitable in cases where the RO has a trust relationship with the client, such as its computer operation system or a highly privileged application since the client must discard the password after using it to obtain the access token.

Refresh Token – A refresh token is often returned with an access token. Once the original access token expires, the corresponding refresh token can be sent to the OAuth AS to obtain a fresh access token without requiring the resource owner to re-authenticate. This allows short-lived tokens to exist between the client and the Resource Server, and long-lived tokens between the client and the AS.

Client Credentials – The client presents its own credentials to the OAuth AS in order to obtain an access token. This access token is either associated with the client's own resources, and not a particular resource owner, or is associated with a resource owner for whom the client is otherwise authorized to act.
Extension Grant Types

OAuth provides an extension mechanism for defining new extension grant types to support additional clients or to provide a bridge between OAuth and other trust frameworks. An OAuth client uses an extension grant type by specifying an absolute URI as the value of the `grant_type` parameter and by adding any additional parameters necessary (see “Token Endpoint” on page 543).

PingFederate supports the following extension grant types:

**SAML 2.0 Bearer** – The client obtains a SAML 2.0 Bearer Assertion and uses it to request an access token from the OAuth AS. This grant type allows a client to use an existing trust relationship, expressed through a SAML assertion, without a direct user approval step at the AS.

**Validation Grant Type** – This proprietary PingFederate OAuth extension enables an RS to act as a client in the request/response exchange with the OAuth AS. The grant type allows an RS to check with the OAuth AS on the validity of a bearer access token received from a client making a protected-resources call.

Persistent vs. Transient Grants

Several grant types are usually regarded as persistent, valid until they are explicitly cancelled. Depending on configuration settings, persistent grants can result from the Authorization Code, Resource Owner Credentials, and Implicit grant types (see “Configuring a Client” on page 178).

Transient grants are valid only for the lifetime of the access token itself. Client Credential access grants, for example, are considered transient.

Support for persistent grants requires PingFederate to use a database for long-term storage. The database contains a table defining a `USER_KEY`, which can be populated according to information mapped using attributes obtained during a user's initial authentication verification within PingFederate (see “Mapping OAuth Attributes”, next).

**Important:** A pre-installed, default Hypersonic database is available for initial setup and testing. However, we strongly recommend that you choose your own, secured database for production deployments. See “Defining an OAuth Grant Data Store” on page 131 for more information.

Mapping OAuth Attributes

For OAuth persistent grants, user attributes are mapped in the administrative console for a two-stage processing flow, as shown in the following illustration. (Note that this mapping is different from other mapping scenarios in
About OAuth

PingFederate, which involve just a one-phase configuration—see “About Attributes” on page 19.)

To accomplish the first stage (top of diagram), mapping is required for setting up persistent grants, including a user key (see the previous section, “Persistent vs. Transient Grants”). The mappings may use attributes obtained during initial authentication events within PingFederate—via SAML assertions (for Service Providers’ IdP connections for SSO), IdP adapters, and/or Resource Owner credentials (see “Using OAuth Menu Selections” on page 156). Data-store lookup is provided as needed (for example, to retrieve an LDAP ID for storage as the user key).

The second mapping configuration involves mapping user keys derived from the first mapping(s) into OAuth access tokens. Data stores may also be used here to retrieve any required user attributes.

At runtime, the first time a client requests an OAuth token the two mapping sequences are employed sequentially. The second mapping is invoked every time a new access token is requested based on an existing persistent grant.

Scopes

Where OAuth provides a mechanism to constrain the privileges associated with an access token, scopes provide a way to more specifically define the privileges requested and granted.

Generally, a client specifies the scopes desired when asking for authorization. The issued access token is associated with these approved scopes. Scopes are configured globally (see “Authorization Server Settings” on page 159) but may be restricted on a client-by-client basis (see “Client Management” on page 177).

OAuth User-Facing Screens

The PingFederate OAuth AS provides two screens presented to end users (resource owners) during OAuth transactions, one requesting approval of the
Key Concepts

scope of protected resources requested and one providing a means of revoking persistent access grants.

Tip: These screens can be customized and branded as needed (see “Customizing User-Facing Screens” on page 87).

OpenID Connect

As an extension of OAuth capabilities, PingFederate supports an optional configuration for OpenID Connect, an emerging protocol for secure, lightweight transfer of authentication and user attributes (see openid.net/connect).

PingFederate supports both the OpenID Connect Basic and Implicit Profiles defined in the standard. In both of these profiles, the end result is the release of at least two tokens to the requesting client application: an ID token and an OAuth access token. (Depending on associated grant types, a refresh token may also be released.)

The ID token is an integrity-secured, self-contained token (in JSON Web Token format) containing claims about the user, namely the subject. A client uses the ID token to securely identify the user authenticated by an OpenID Connect Provider (OP) accessing the client application. A client may subsequently use the OAuth access token to retrieve additional claims about the user, such as a complete profile containing full name, email, phone and other schema elements defined in the OpenID Connect specifications (see “Configuring OpenID Connect Policies” on page 170).

A UserInfo endpoint needed for the client to receive additional attributes with the access token is available as a standard OAuth-protected endpoint (see “OAuth 2.0 Endpoints” on page 543).

To configure policies, first enable the protocol in Server Settings (see “Choosing Roles and Protocols” on page 106). To make use of OpenID Connect defined scopes for accessing various degrees of user-associated claims, you must also configure them as allowed Scope Values in the OAuth AS (see “Authorization Server Settings” on page 159).

Furthermore, PingFederate provides a front-channel endpoint for OAuth Clients using the OpenID Connect protocol to close other associated sessions and a back-channel Web service for clients to revoke end-user sessions. For more information, see “Centralized Session Management” on page 157.

SSO Integration Kits and Adapters

As a stand-alone server, PingFederate must be programmatically integrated with end-user applications and identity management (IdM) systems to complete the “first- and last-mile” implementation of a federated identity network for browser-based SSO.

Note: See the PingFederate SSO Integration Overview for more information.

For an IdP (the first mile), this integration process involves providing a mechanism through which PingFederate can look up a user’s current
authenticated session data (for example, a cookie) or authenticate a user without such a session. For an SP, the last mile involves enabling PingFederate to supply information needed by the target application to set a valid session cookie or other application-specific security context for the user.

To enable both sides of this integration, PingFederate provides bundled and commercial integration kits, which include adapters that plug into the PingFederate server and agent toolkits that interface with local IdM systems or applications, as needed.

In addition, for IdP deployments PingFederate provides plug-in adapter selectors, which enable dynamic selection of adapter instances based on administrator-specified criteria (see “Bundled IdP Adapter Selectors” on page 15).

PingFederate also includes a robust software development kit (SDK), which software developers can use to write their own adapters for specific systems. Adapters can be written to retrieve attributes from custom data stores, connect to application- or IdM-specific user authentication systems, or provide complex attribute transformations or processing.

**Bundled Adapters**

PingFederate packages these adapters:

- An OpenToken Adapter, which provides a generic interface for integrating with various applications, including Java- and .NET-based applications (see “OpenToken Adapter Configuration” on page 499)
- An IdP Composite Adapter, which allows multiple configured adapters to execute in sequence for browser SSO. This capability, called adapter chaining, may be used either for single-adapter usage, depending on authentication context, or to support multi-factor authentication via a series of adapters.
- HTTP Basic and HTML Form IdP Adapters, which are used in conjunction with Password Credential Validators (see “Validating Password Credentials” on page 217). These adapters provide integration, for example, with LDAP user-data stores such as Active Directory or direct user logon via credentials maintained by PingFederate. (See “Configuring the HTTP Basic IdP Adapter” on page 509 and “Configuring the HTML Form IdP Adapter” on page 515.)

**Bundled IdP Adapter Selectors**

For an IdP, PingFederate also includes global (cross-connection) adapter selectors. Along with the Composite Adapter (see the previous section) and token authorization, the selectors enable dynamic integration with an organization’s authentication or authorization policies (also known as adaptive federation).

**Tip:** The results of adapter-selector criteria evaluation may be used to select subsequent selectors, which allows handling of complex hierarchical access-policy decisions (see “Mapping Selector Results to Adapter Instances” on page 242).

**CIDR Selector** – Provides a means of choosing adapters at runtime based on whether an end-user’s IP address falls within a specified range, or ranges (using Classless Inter-Domain Routing notation). This selector allows administrators to
Key Concepts

determine, for example, whether an SSO request originates inside or outside the corporate firewall and use different authentication integration accordingly (see “Configuring the CIDR Adapter Selector” on page 237).

**Cluster Node Selector** – Provides a means of choosing adapters at runtime based on the PingFederate cluster node that is servicing the request. For example, this selector allows you to choose whether or not Integrated Windows Authentication is attempted based on the PingFederate cluster node with which a Key Distribution Center is associated (see “Configuring the Cluster Node Selector” on page 237).

**Connection Set Selector** – Provides a means of selecting an adapter at runtime based on a match found between the target SP connection used in an SSO request and SP connections configured within PingFederate. For example, administrators with different requirements for SP connections can override connection adapter selection on an individual connection basis (see “Configuring the Connection Set Selector” on page 238).

**HTTP Header Selector** – Provides a means of choosing an adapter at runtime based on a match found (using wildcard expressions) in an HTTP header. This selector allows administrators to determine, for example, authentication behavior based on the type of browser (see “Configuring the HTTP Header Selector” on page 238).

**OAuth Scope Selector** – Provides a means of choosing an adapter at runtime based on a match found between the scopes of an OAuth authorization request and scopes configured in the PingFederate OAuth Authorization Server (AS). For example, if a client requires write access to a resource, administrators can configure the selector to choose an adapter that offers a stronger form of authentication such as the X.509 client certificate rather than username and password (see “Configuring the OAuth Scope Selector” on page 239).

**Requested AuthN Context Selector** – Provides a means of picking an adapter at runtime based on the authentication context requested by an SP, for SP-initiated SSO (see “Browser-based SSO” in the “Supported Standards” chapter of Getting Started). Configured adapters are mapped either to SAML-specified contexts or any ad-hoc context agreed upon between the IdP and SP partners (see “Configuring the HTTP Header Selector” on page 238).

**Commercial Adapters and Selectors**

Ping Identity regularly develops and maintains integration kits, including adapters, to work with applications and leading identity management systems. Available kits may be downloaded from the Ping Identity Web site (www.pingidentity.com/support-and-downloads). Adapter Selectors may also be added to the download site periodically; contact your Ping Identity sales representative if you are looking for specific adapter-selection capabilities.

**Software Development Kit**

The PingFederate SDK provides a flexible means of creating custom adapters to integrate federated identity management into your system environment. See the PingFederate SDK Developer's Guide.
Hierarchical Plug-in Configurations

PingFederate allows you to use a configuration of an adapter (as well as certain other PingFederate plug-ins) as a parent instance from which you can create child instances. You can then modify the inherited configuration for the child instances as needed. This feature provides easier management of adapter settings in cases where only small changes to an existing adapter (or plug-in) configuration need to be made for a particular use case.

For example, different SP-connection adapter instances might have their own IdP logon URLs (for branding or other application ownership reasons) while the majority of the other adapter configuration settings are the same. In this case, you might want to use a parent/child configuration to override the logon URLs.

Tip: You can also override adapter instances as part of mapping them into either SP or IdP connections, for cases where overridden settings may apply only to one particular connection configuration.

Any changes to a parent configuration are propagated to its child (or connection-based) configurations provided the changes are not already overridden in the derived instance.

In addition to adapters, PingFederate allows you to create parent/child configurations for the following plug-in types:

- Token Translators (see “Token Processors and Generators” on page 7)
- Password Credential Validators (see “Validating Password Credentials” on page 217)
- Identity Store Provisioners (see “Configuring Identity Store Provisioners” on page 344)

Identity Mapping

Identity mapping is at the core of identity federation. One of the primary goals of SAML is to provide a way for an identity provider (IdP) to send a secure token (the assertion) containing user-identity information that a service provider (SP) can translate, or map, to local user stores. (For more information about SAML, see the “Supported Standards” chapter in Getting Started.)

For browser-based SSO, PingFederate enables two modes of identity mapping between domains:

- Account Linking
- Account Mapping

For WS-Trust STS, account mapping is used.

Account Linking

Under the standards, account linking can be used for browser-based SSO in cases where each domain maintains separate accounts for the same user. Account linking uses the SAML assertion to create a persistent association between these distinct user accounts. The account link, or name identifier, may be either a
unique attribute, such as an email address, or a pseudonym generated by the IdP to uniquely identify individual users. Pseudonyms can be used when privacy is a concern; they cannot easily be traced back to a user's identity at the partner site.

During the user's first SSO request, the SP prompts for local credentials, which enables the SP to link the name identifier contained within the assertion—either an open attribute or a pseudonym—with the user's local account. Subsequent SSO events will not prompt the user to authenticate with the SP, since the SP federation server keeps a table associating remote users' name identifiers with local user accounts. The SP associates the link to the user's corresponding local account and provides access to the account without separate authentication.

**Tip:** An SP PingFederate uses a default, Hypersonic database to handle account linking. You can use your own data store instead, as needed. For more information, see “Configuring an LDAP Connection” on page 122.

Optionally, additional attributes may be sent with the name identifier. When a pseudonym is used as the account link, however, care must be taken to send only general attributes (a user's organizational role or department, for example) that will not compromise privacy.

### Linking Permission and “Defederation”

The SAML specification also allows the SP application to build in user verification and approval of account linking and provides a means for the user to permanently cancel the linking, known as defederation (see “/sp/defederate.ping” on page 537). A user who has defederated may later elect to reassociate with a local user account.

### SP Affiliations

Under the SAML 2.0 specifications, an IdP can configure PingFederate to enable a group of SPs—an SP affiliation—to share the same persistent name identifier (see “Defining SP Affiliations” on page 329). This capability facilitates the use case in which a number of business partners have an existing relationship and sharing a single name identifier among all parties reduces the federation integration effort.

### Account Mapping

**Account mapping** (also called “attribute mapping”) enables an SP to use PingFederate to perform a user lookup and map a user's identity dynamically based on one or more attributes received in the assertion. The attributes used to look up the user are always “exposed”; that is, they are known to both the IdP and SP. An email address, for example, is a commonly used identifying attribute.

Account mapping can be used to achieve one-to-one mapping (individual user accounts exist on both sides of federated connection) or many-to-few (IdP users without accounts at destination sites may be mapped to guest accounts or to a role-based general account).

For browser-based SSO, transient identifiers provide an additional level of privacy—virtual anonymity—by generating a different opaque ID each time the
user initiates SSO. Transient IDs are often used in conjunction with federation role mapping, whereby the user is mapped to a guest account or to a role-based account based on the user’s association with the IdP organization rather than personal attributes.

As with pseudonyms, additional attributes may be sent with the transient identifier. Again, care should be taken to preserve privacy.

Account mapping is commonly implemented in B-to-B or B-to-E use cases where it might be appropriate for the administrator to create a user lookup on behalf of the user.

About Attributes

Federation transactions require, at a minimum, the transmission of a unique piece of information (such as an email address) that identifies the user for identity mapping between security domains.

In addition to attributes used for identity mapping, the IdP can pass other user attributes in an assertion (including SAML tokens for Web Services). This supplemental information can be used by the SP for several purposes. For example, attributes may be used to map and authorize the user into a specific role, with associated site permissions. In other cases, attributes may be used to customize the end application display for a more robust user experience.

The SP also has the option of incorporating additional attributes prior to creating a session for the target application. This is commonly done where the SP also maintains an account for the user and wants to pass additional information for profiling or access-policy purposes.

Attributes must be carefully managed between IdPs and SPs. PingFederate facilitates the process by providing configuration steps that enable administrators to:

- Define and enforce attribute contracts for each partner connection.
- Define and retrieve attributes from the adapter or STS token processor to populate an attribute contract directly or use these attributes to look up additional attributes in IdP data stores.
- Define and enforce a set of required attributes needed by SP adapters or STS token generators to interface local systems or applications (see “Adapter Contracts” on page 22).
- Set up connections to local data stores (see “Data Stores” on page 23).
- Configure specific attribute sources and lookups—based on the data stores—and map attributes into IdP assertions or into SP adapters or token generators used to interface target applications (see “SSO Integration Kits and Adapters” on page 14 or “Token Processors and Generators” on page 7).
- Selectively mask attribute values recorded in transaction logs (see “Attribute Masking” on page 25).

Attribute Contracts

An attribute contract represents an agreement between an SP and an IdP about user attributes sent in a SAML assertion, either for browser-based SSO or WS-
Trust STS. The contract is a list of case-sensitive attribute names. IdPs and SPs must configure attribute contracts to match.

**Tip:** When privacy is required for sensitive attributes, you can configure PingFederate to mask their values in log files (see “Attribute Masking” on page 25).

For an IdP, the attribute contract defines which attributes PingFederate sends in an assertion. While this contract is fixed for all users authenticating to the SP partner, the values used to fulfill the contract may differ from one user to the next. The attribute contract may be fulfilled by relying on a combination of different data sources:

- The IdP adapter or STS token processor
- An IdP attribute source, which identifies the location of individual attributes in a data store
- Static text values for some attributes, or text values combined with variables
- Expressions (see “Using Attribute Mapping Expressions” on page 573)

For an SP, the attribute contract defines the attributes PingFederate expects in a SAML assertion. PingFederate can be configured to pass these attributes to the SP adapter or, for Web Services, to the SP token generator (see “Configuring SP Adapters” on page 338 or “Configuring Token Generators” on page 476). You can also use attributes to look up additional attributes in local data stores, which may be needed to start a user session or create a local security token for Web Services (see “Adapter Contracts” below or “STS Token Contracts” on page 23).

The attribute contract must contain the attribute SAML_SUBJECT, the primary information used to identify the user, unless you are using account linking for browser-based SSO. This attribute is automatically included when creating a new contract.

**Note:** You create attribute contracts on a per-connection basis. For example, if an SP has deployed two session-creation adapters for two separate applications, a single attribute contract can be created for the IdP connection partner. This single contract would be constructed to supply all the attributes needed by both SP adapters.

**Name Formats**

By agreement with an SP partner, an IdP may specify a format (email, for example) associated with the SAML_SUBJECT. The SP may require this information to facilitate handling of the format.

The partner agreement may also include a requirement for the IdP to provide format specifications associated with other attributes.

For browser-based SSO connections, PingFederate provides a means for an IdP administrator to select from among standard subject and/or attribute formats, depending on the relevant SAML specifications. An administrator can also
define a customized selection of additional attribute formats (see “Creating an Attribute Contract” on page 264).

**Note:** The designation of formats is not applicable to SP administrators. The information is simply available in the incoming assertion to an SP application that might need it for particular processing requirements.

For the WS-Trust IdP configuration, attribute-name formats cannot be specified. If needed, however, an administrator can use a special variable in the attribute contract to set the subject-name format (see “Defining an STS Attribute Contract” on page 454). (The same variable is also available for browser-based SSO attribute contracts, but the feature is deprecated.)

**STS Namespaces**

By agreement with an SP partner for a WS-Trust STS connection, an IdP may specify an XML namespace to be associated with an attribute (for example, to use claims-based authorization with WIF clients—see “Windows Identity Foundation Clients” on page 8). Namespaces can be specified only for attributes of a WS-Trust IdP configuration using SAML 1.1 as the default token type (see “Defining an STS Attribute Contract” on page 454).

**Adapter Contracts**

An adapter contract represents an agreement between the PingFederate server and an external application. In concert with the attribute contract between partners, adapter contracts specify the transfer of attributes. Adapter contracts consist of a list of case-sensitive attribute names.

On the IdP side of a federation, adapter attributes are supplied to PingFederate by an IdP adapter (see “SSO Integration Kits and Adapters” on page 14 and “Configuring IdP Adapters” on page 230).

On the SP side, adapter contract attributes are those required by an adapter to start a session with an application. At least one adapter type is needed for each security domain. Then an adapter instance must be configured for each target application. (See “Configuring SP Adapters” on page 338.)

Adapter contracts on the SP side are fulfilled using attributes from the attribute contract, possibly enhanced through other attributes looked up from local data stores. For example, if several target applications are controlled by the same security context and can receive the same set of attributes to start a session for the user, you would deploy an adapter type and configure an adapter instance for each protected application (see “Configuring Adapter Mapping and User Lookup” on page 370).

**Extended Adapter Contract**

Adapter contracts are created when an adapter type is deployed with PingFederate. When developed, these adapters are “hard-wired” to look up or set a specific set of attributes. After deployment, your attribute requirements may change. To streamline adjustment of adapter contracts, PingFederate allows an administrator to add additional attributes to the adapter instance through the administrative console. These adjustments are called extended adapter contracts.
STS Token Contracts

Similar to an adapter contract for browser-based SSO, an STS token-processor or token-generator contract represents an agreement between the PingFederate server and an external application in the context of a Web Services transaction. In concert with the attribute contract between partners, token contracts specify the transfer of attributes, consisting of a list of case-sensitive attribute names.

On the IdP side of a federation, token-processor attributes are supplied to PingFederate (see “Token Processors and Generators” on page 7 and “Configuring Token Processors” on page 443).

On the SP side, token-generator contract attributes are those required by a token generator to pass identity information from the token to the Web Service client application. At least one token generator type is needed for each security domain. Then a token generator instance must be configured for each target application (see “Configuring Token Generators” on page 476). If several target applications are controlled by the same security context and can receive the same set of attributes for the user, you would deploy a token generator type and configure a token generator instance for each target application (see “Mapping Token Generators” on page 482).

Extended Token Generator Contract

Token-generator contracts are created when a token-generator type is deployed with PingFederate. When developed, these token generators are “hard-wired” to look up or set a specific set of attributes. After deployment, your attribute requirements may change. To streamline adjustment of token-generator contracts, PingFederate allows an administrator to add additional attributes to the token-generator instance through the administrative console. These adjustments are called extended token-generator contracts.

Data Stores

PingFederate can be configured to use local data stores to supply attributes for either the IdP’s attribute contract, the SP’s adapter contract, or STS token contracts (see sections above). Standard data stores may include any JDBC-accessible database or an LDAP v3-compliant directory server (see “Managing Data Stores” on page 117).

Alternatively, you can use the PingFederate Custom Source SDK to create your own driver for non-JDBC/LDAP data stores—including, for example, flat files or SOAP-connected databases (see the PingFederate SDK Developer’s Guide).

Data stores can be used across multiple connections.

Multiple Data Source Attribute Mapping

When querying data stores for attributes to help fulfill the attribute contract, PingFederate can be configured to use more than one attribute source when mapping values to an assertion. For browser SSO for an IdP, multiple data stores can be used in two ways:

- Set up search parameters separately for each data store and for “fall-through” searches. For example, you can add the same data store more than once, using different search queries for each instance, or you can search different
data stores successively. If any search fails to find a user in the specified attribute source, the next search is executed until a match is found. A failsafe attribute source can also be configured, providing a default set of attributes from the IdP adapter and using text values (see “Attribute Source Setup” on page 272).

- Configure separate data stores to look up attributes for a single mapping. For example, you can query multiple data stores for values and map those values in one mapping, or query a data store for a value and use that value as input for subsequent queries of other data stores (see “Attribute Source Setup” on page 272).

Note: SSO fails if the user is not found in any data source.

Attribute Masking

At runtime PingFederate logs user attributes (see “Managing Log Files” on page 44). To preserve user privacy, you may wish to mask the values of logged attributes.

PingFederate provides this masking capability at all points where the server logs attributes. These points include:

- Data-store lookup at either the IdP or SP site (see “Managing Data Stores” on page 117).
Chapter 1
Key Concepts

- Retrieval of attributes from an IdP adapter or token processor (see “Setting Pseudonym Values and Masking” on page 234 and “Setting Attribute Masking” on page 449).

- SP-server processing of incoming attributes based on the SSO attribute contract, (see “Defining an Attribute Contract” on page 369).

  Note that the SAML Subject ID is not masked: the SAML specifications provide for either pseudonymous account linking or transient identification to support privacy for the Subject ID (see “Account Linking” on page 18).

- SP-server processing of incoming attributes in response to an Attribute Request under XASP (see “Defining Security Policy” on page 402).

  For information about XASP, see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started.

---

**Important:** Many adapter implementations, as well as other product extensions, may independently write unmasked attribute values to the PingFederate server log. These implementations are beyond the control of PingFederate. If sensitive attribute values are a concern when using such a component, a system administrator can adjust the component’s logging threshold in log4j.xml to prevent the recording of attributes (see “Managing Log Files” on page 44).

---

About Token Authorization

PingFederate provides an optional configuration to evaluate user attributes, as well as other runtime variables (such as authentication context), for authorization purposes. This feature, known as token authorization, provides a way for administrators to extend access policy directly to browser SSO, STS, and OAuth events by conditionally allowing or disabling the issuance of relevant security tokens (for example, SAML assertions, STS tokens, OAuth access tokens, or session cookies). The option is also available for extending authorization policy to attribute-query responses (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

Administrators can configure token authorization using Issuance Criteria screens immediately following the configuration of attribute mapping at all applicable points in the administrative console (see, for example, “Specifying Issuance Criteria (Optional)” on page 285). The option is available for any IdP, SP, or OAuth configuration, or for direct SSO adapter-to-adapter or STS to-to-token setups.

Issuance Criteria

The token-authorization configuration consists of one or more rules that evaluate attribute values against selected conditions (see “Single and Multi-Value Conditions” below). You can choose from among several sources for the attributes, depending on the type of configuration that contains the token-authorization setup. The sources always consist of all of those available for attribute mapping, including data stores (when configured) and runtime information related to the context of an event. In addition, values of mapped attributes are available to provide access to any plain-text mappings or the
runtime results of any expression mappings (see “Using Attribute Mapping Expressions” on page 573).

**Tip:** When more than one condition is configured, all are evaluated and all the conditions must be met at runtime (evaluated as “true”) for authorization to succeed and processing to continue. In cases where you might need “or” conditions or layered evaluations, you can create one or more OGNL expressions using an Advanced Criteria feature. For more information, see “Using Attribute Mapping Expressions” on page 573 and “Expression Examples” on page 576.

**Note:** When authorization fails and a transaction is halted, a configurable Error Result code is passed back up through the system, potentially to an application layer or as a variable on a PingFederate user-facing template. How this code is interpreted depends on the use case and application integration. For more information, see Error Result descriptions in sections of this document covering specific token-authorization configurations.

**Single and Multi-Value Conditions**

Each token-authorization configuration provides a choice of conditions for evaluating attribute values:

Use one of the first four choices only for attributes consisting of a single value.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

**Security Infrastructure**

This section describes the PingFederate security infrastructure that supports encrypted messaging, certificates, and digital signing. These functions are integrated into PingFederate’s configuration screens to provide complete control
Digital Signatures

A digital signature is a way to verify the identity of a person or entity who originates an electronic document and ensure that the message has not been altered. Digital signatures are used in both SAML (including STS tokens) and WS-Federation electronic documents.

Handling a digital signature involves message signing, signature and certificate validation, and signing-policy coordination between connection partners.

Message Signing

Certificates contain information about the owner of the certificate along with a public key. Applying a digital signature creates and encrypts a hash from the message you are signing, using your private key. PingFederate provides a choice of signature encryption algorithms when a stronger algorithm is required.

To ensure the integrity of SAML messages or STS tokens, we recommend digital signing practices using public/private keypairs in conjunction with X.509 certificates.

Note: Digital signatures do not encrypt the contents of a message; SSL/TLS and/or XML encryption is used for this purpose.

The certificate should be signed by a Certificate Authority (recommended), but it can be self-signed or signed by an untrusted third party. After generating a keypair and a self-signed certificate, you can use PingFederate to create a Certificate Signing Request (CSR) and send it to a CA for signing. After the CA has generated a Certificate Signing Response, you can import it into PingFederate's certificate management system. (The CA's certificate must be in PingFederate's trusted store or in the Java runtime cacerts store.)

PingFederate enables signing and validation of responses, requests, and/or the assertion message. In addition, PingFederate provides for certificate generation, import and export functionality, CSR generation, and application of digital signatures. You can create reusable global signing certificates across your federated connection base and import signature verification certificates for each partner (see “Digital Signing and Decryption Keys and Certificates” on page 209).

Note: Ping Identity recommends generating unique certificates for each connection, which limits exposure if the private key becomes compromised.

Signature Validation

After receiving a signed message, PingFederate verifies the signature using the public key that corresponds with the private key used to sign the message or token. Verification involves creating a hash of the received message, using the signing partner's public key to decrypt the hash sent with the original message, and verifying that both hash values are equal.
Certificate Validation

PingFederate always checks certificates to see if they have expired, both when they are initially imported and at runtime when they are used to encrypt, decrypt, and digitally sign or verify assertions.

PingFederate can also check to see whether a certificate has been revoked, using either Certificate Revocation Lists (CRLs) or the Online Certificate Status Protocol (OCSP). Depending on the content of the certificate in question and your requirements, the server will perform either of these checks during SSO or SLO processing for the following cases:

- Signature verification
- Validation of a client certificate used for authentication to PingFederate when the server is handling direct client requests
- Validation of the server SSL certificate when PingFederate is acting as the client making an HTTPS request to a separate server

If a certificate is expired or revoked, the associated SSO or SLO transaction fails at runtime and an error is written to the transaction log. In the administrative console, an expired or revoked certificate is identified as such in the Status column of its respective Certificate Management list.

CRL Revocation Checking

This process involves querying a CRL distribution-point URL and ensuring that a certificate is not on the returned revocation list maintained at the site. The URL is specified in the certificate.

No setup is needed in the administrative console to enable CRL checking. PingFederate automatically checks CRLs if all of the following conditions are met:

- The certificate contains the URL where the CA maintains its CRL.
- The URL is accessible.
- The returned CRL is signed and the signature verified.
- CRL validation is not explicitly disabled as a failover option in the OCSP setup (see “Certificate Revocation Checking” on page 212).

OCSP Revocation Checking

OCSP was developed as an alternative to CRL validation and provides a more centralized and potentially more reliable means of checking certificate status. In this scenario, an OCSP Responder URL is normally embedded in the incoming certificate (a configured default URL may be used, alternatively). The URL, maintained by the issuing CA, is used to query the certificate status.

The primary difference between OCSP and CRL checking is how the verification occurs. CRL checking requires the requesting client to determine if the certificate has been revoked (or if any of the certificates in the chain of issuer certificates has been revoked), based on the returned CRL. With OCSP, the client sends the certificate itself, and revocation checking is handled by the Responder server, which returns the certificate status.

A PingFederate administrator can enable and configure OCSP processing in the administrative console (see “Certificate Revocation Checking” on page 212). The protocol may be used exclusively or in conjunction with CRL checking as a backup.
For more information about OCSP, see www.ietf.org/rfc/rfc2560.txt.

**Digital Signing Policy Coordination**

To coordinate digital signature policy, partners must first agree about whether they will sign SAML messages or tokens. In some cases, the protocol specifications require signatures—for example, all SAML STS tokens and all SSO assertions sent across the POST binding must be signed. (These requirements are enforced by the PingFederate administrative console and the runtime protocol engine.) Other uses of the digital signatures are optional between partners and enforced if specified for a partner connection.

If a digital-signing certificate is not issued by a trusted CA (that is, “self-signed”), then the signing partner must send the public-key certificate out-of-band (for example, via email) to the partner. The partner must import the certificate into PingFederate when configuring a connection to the signing partner for SSO/SLO or STS.

If the certificate is signed by a trusted CA and the signing partner chooses to embed the certificate in all signed messages, then the verifying partner can elect to use the embedded certificate for signature verification, after validating it against the Subject DN of the original certificate. The public-key certificate may or may not be sent out-of-band (just the Subject DN is required).

**Tip:** PingFederate can extract the Subject DN from the certificate, when available, during the signature-verification configuration.

The next section provides more information about the two alternative signature-verification trust models described above, from the standpoint of the verifying partner.

**Trust Models**

For validating digital signatures, PingFederate provides a selection of trust models in the administrative console for each partner connection, based on the certificate categories listed below. Note that for each trust model, PingFederate always verifies that the certificate is current and that the signature in the message can be verified using the certificate specified. Additional checks depend upon the trust model selected.

- **Anchored Certificate**

  In this case, certificates used for signature verification must be issued by a trusted CA, and the certificate chain must be verifiable recursively back to the root issuer. PingFederate validates the certificate (including recursive revocation checking, when enabled, back to the issuer) for all signed messages from the partner.

  In addition, when this option is selected, the incoming message must include the verification certificate for the signature. PingFederate uses that certificate to verify signatures from the partner if its Subject DN matches the partner’s public certificate, as specified in the administrative console, and the issuer CA certificate is part of the trusted store. This feature provides a dynamic trust model that overcomes the problem of interrupting service to change out expired certificates.
• Unanchored Certificate

When this option is chosen, incoming signatures are verified exclusively using a specific certificate imported for a connection into PingFederate (or a secondary, backup certificate when specified). The certificate may be self-signed or issued by a trusted CA. The certificate chain, if any, is not verified. However, revocation checking, when enabled, is performed up any existing chain as far as available.

Secure Sockets Layer

SSL certificates signed by a CA can be used to identify one or both ends of the federation. SSL/TLS provides an encrypted connection between the two parties in which the content of a message is not exposed, thus ensuring confidentiality and message integrity.

SAML SSL and TLS Scenarios

SSL/TLS should be used in association with the SOAP responder URL and Single Sign-on Service located at an IdP site. On the SP side, the Artifact Resolution Service should also use SSL/TLS. Optionally, SSL/TLS may also be used to secure communication between internal data stores and PingFederate and between the PingFederate STS and Web Service client or provider applications.

Authentication

Three methods of authentication, described below, are available for use with PingFederate for browser-based SSO to authenticate connection partners making SOAP requests. For SOAP authentication by STS clients, a separate option using either or both of the first two methods, may be configured (the third method, digital signing, is automatically required). The selection of one or more method(s) must be agreed upon between partners and synchronized within IdP and SP federation implementations:

• HTTP Basic Authentication: partners identify themselves by passing username and password credentials.

• SSL Client Certificate Authentication: partners use SSL Client Certificates presented during SOAP request transactions. Each partner needs to import
the other’s certificate out-of-band (see “SSL Client Keys and Certificates” on page 207).

**Important:** The SSL/TLS server-client handshake involves negotiating cipher suites to be used for encryption/decryption on each side of a secured transaction. PingFederate supports only stronger cipher suites; to enhance security, weaker cipher suites are commented out of two configuration files located in `<pf_install>/server/default/data/config-store:
com.pingidentity.crypto.SunJCEManager.xml
com.pingidentity.crypto.NcipherJCEManager.xml`

To ensure the most secure transactions, we recommend that administrators retain this cipher-suite configuration.

Due to import control restrictions, the standard Java Runtime Environment (JRE) distribution supports strong but not unlimited encryption. For this reason, the strongest cipher suites in the same configuration files are also commented out. To use the strongest encryption, when permissible, remove the comments from the AES 256 cipher suites and download and install the appropriate version of “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” from the Oracle download Web site (www.oracle.com/technetwork/java/archive-139210.html).

- Digital Signatures: partners sign the XML message transmitted via the SSL/TLS connection. Signatures are verified by the receiver based upon the certificate(s) configured for that connection. Each partner should import the other’s certificate(s) out-of-band (see “Digital Signing and Decryption Keys and Certificates” on page 209).

**Verifying Trusted Certificates**

PingFederate validates the trust of all certificates. A certificate is trusted if the certificate of its issuer is in PingFederate’s trusted certificate store. The root certificate of the CA, by which a certificate is issued, must be imported into PingFederate’s trusted certificate store or contained in the Java runtime cacerts store.

**XML Encryption**

PingFederate supports the optional SAML 2.0 specification allowing for encryption of assertions (including STS SAML tokens), which further enhances confidentiality when required.

For SAML 2.0 SSO connections you can choose to encrypt entire assertions, the user’s name identifier, and/or other user attributes. You can use signature verification and signing keys to encrypt and decrypt messages, respectively.

**Using Auto-Connect**

PingFederate allows organizations to provide secure SSO on the fly—that is, without the need for configuring partner-specific, browser-based SSO.
connection parameters. This feature—Auto-Connect™—extends SAML 2.0 SP-initiated SSO or SLO and metadata specifications to enable deployments to retrieve partner connection information securely on an as-needed basis. (For information about SAML 2.0, see the “Supported Standards” chapter in Getting Started.)

The feature is especially useful to an SP who wants to provide SSO capability to more than one partner. A SaaS provider, for example, can provide SSO to innumerable clients without specifying redundant connection information for each one. Auto-Connect can also help an enterprise, acting as an IdP, to provide easily scalable SSO for multiple outsourced services.

For either an IdP or SP PingFederate server, you can implement Auto-Connect for any number of partners by configuring a common initial setup and a list of domain names. For an IdP, the domain-name list contains SP partners from whom your site will accept Auto-Connect authentication requests. For an SP, the list contains IdP-partner domains to which your site can send authentication requests and receive SSO assertions.

For information about configuring Auto-Connect for your federation partners, see “Configuring SP Auto-Connect” on page 332 or “Configuring IdP Auto-Connect” on page 433.

Providing Metadata

You enable Auto-Connect as part of Server Settings from the Main Menu (see “Choosing Roles and Protocols” on page 106). Once Auto-Connect is enabled and the initial setup is fully configured and activated, partners can retrieve your connection metadata via HTTP. At runtime, Auto-Connect deployments at partner sites use the endpoints provided in the metadata to interact with your server and complete SSO or SLO processing.

The metadata, which follows SAML 2.0 specifications, must be signed, and the validity of the data is time-limited (see “Auto-Connect Security Model” on page 34 and “Configuring Auto-Connect Metadata Lifetime” on page 116).

Runtime Processing

Auto-Connect runtime processing starts when a user tries to reach a protected SP resource. The process depends on SP Web-application functionality that determines the user’s IdP domain (for example, from a submitted email address) and passes it to the SP PingFederate server in the SSO request.

Figure 1 and the accompanying “Processing Steps” describe the complete SSO processing flow:
Chapter 1

Key Concepts

Figure 1: Auto-Connect Processing Flow

Processing Steps

1. User sends a logon request with an email address to an SP application. For example:
   john@mycompany.com

2. The application parses the email address and sends a request to PingFederate. For example:
   https://hostname.com:9031/sp/startSSO.ping/?Domain=mycompany.com

3. The SP PingFederate server looks up the domain in a list of domain names allowed to use Auto-Connect.

4. If the domain is in the list, the SP retrieves connection metadata from the IdP's public endpoint.
   By default, PingFederate looks for the metadata by prepending http://saml to the domain. For example:
   http://saml.mycompany.com
   This default location can be changed, if necessary, in the Allowed Domains lists configured in the PingFederate administrative console.

5. After validating the metadata (see “Auto-Connect Security Model” on page 34), the SP sends an authentication request to the IdP's SSO service.

6. If the request <Issuer> is not among the IdP's static-connection partners, the IdP PingFederate server looks for the issuer's domain name in the list of domains allowed to use Auto-Connect.

7. The IdP retrieves the SP's metadata via its public endpoint and verifies the metadata signature.
   The process is the same as that used by the SP in Step 4.

8. The IdP requests user authentication via the configured adapter instance.
9. Once the user is authenticated, the IdP returns a signed SAML assertion to the SP’s Assertion Consumer Service (ACS) endpoint.

10. (Not shown) The SP logs the user on to the requested resource via the configured SP adapter.

## Auto-Connect Security Model

Auto-Connect processing requires digital signatures to ensure the authenticity of the published metadata as well as all subsequent SSO or SLO requests and responses. The certificate used to sign the metadata is included in the metadata, and all certificates must be signed by a trusted Certificate Authority; thus, partners need not exchange certificates out of band.

In addition to validating certificates, the PingFederate runtime server compares the partner certificate with the entity ID (the “Issuer”) found in the SAML message. Then the server matches the entity ID against the configured list of allowed Auto-Connect domains.

Figure 2 illustrates the security validation process:

![Auto-Connect Security Model Diagram](image)

**Figure 2:** Auto-Connect Security Model

Note that the diagram assumes that the same certificate is used for signing both the metadata and the runtime SAML messages. This is convenient, but not required.

## User Provisioning

PingFederate provides cross-domain user provisioning and account management. User provisioning is an important aspect of identity federation. Often when organizations enable SSO for their users, they must ensure that some form of account synchronization is in place. Automated user provisioning features within PingFederate free administrators from having to devise a manual strategy for this.
Provisioning support takes different forms, depending on what role PingFederate plays in an identity federation, and may be configured either in conjunction with partner SSO connections or separately:

- At an IdP site, you can automatically provision and maintain user accounts at service-provider sites that have implemented the System for Cross-domain Identity Management (SCIM), or at selected SaaS providers (see the next section, “Outbound Provisioning for IdPs”).

  For information about SCIM, see the Web site www.simplecloud.info.

- When PingFederate is configured as an SP, you can provision and manage user accounts for your own organization automatically, by using the standard SCIM protocol or by using identity information received during SSO events from SAML assertions (see “Provisioning for SPs” on page 36).

### Outbound Provisioning for IdPs

For IdP sites, PingFederate provides built-in automated provisioning and user-account management to SCIM-enabled services providers and to selected SaaS providers, via their proprietary provisioning APIs.

**Tip:** Support for provisioning for SaaS applications, including quick-connection templates for partner SSO—SaaS Connectors—is available separately from Ping Identity. Contact sales@pingidentity.com for more information.

Outbound Provisioning also provides an automated means of account disabling or deprovisioning, which may be of key importance to system auditors.

When Outbound Provisioning is enabled, the PingFederate runtime engine polls the IdP organization’s user store periodically (see “Choosing a Connection Type” on page 252). The server uses a separate database internally to monitor the state of the user store and keep user data synchronized between the organization and the target service provider (see Figure 3).

![Figure 3: Outbound Provisioning Processing](image)

As user-data sources, PingFederate provides built-in support for Microsoft’s Active Directory (AD) and Oracle Directory Server (formerly Sun Directory Server); templates are used to preconfigure many provisioning settings. Although these are the only data stores formally tested and supported, other LDAP data stores will likely work as well (see “Identifying the Source Data Store” on page 319). For convenience, PingFederate provides a sample template that can be used for other types of LDAP servers to simplify the provisioning configuration (see “Configuring an LDAP Connection” on page 122).
Tested internal data stores used for synchronization include Hypersonic, MySQL, MS SQL Server, and Oracle databases (a demonstration-only, embedded Hypersonic database is installed by default). Again, any relational database may be used—scripts are provided to aid setup (see “Configuring Outbound Provisioning Settings” on page 114).

For more information, see “Configuring Outbound Provisioning” on page 315.

Provisioning for SPs

When PingFederate is enabled as an SP, two provisioning choices are available:

- **Inbound Provisioning** for SCIM requests coming either from inside or outside an organization
- **Just-in-Time Provisioning** for creating and updating user accounts based on information contained in SAML assertions

**Inbound Provisioning**

Inbound Provisioning provides support for incoming SCIM messages containing requests to create, read, update, or delete/deactivate user records in Microsoft AD data stores. An administrator can configure this provisioning feature by itself or in conjunction with an SSO or other connection type (see “Connection Types” on page 5).

In effect, Inbound Provisioning provides an organization with a dedicated SCIM service provider, which can route user-management requests to an organization's centralized user store. The requests may originate from trusted applications within an organization—for example, a human-resources onboarding SaaS product—or from trusted partner IdPs.

For setup information, see “Configuring Inbound Provisioning” on page 413.) For application-development information about using PingFederate endpoints for SCIM provisioning, see “SCIM Inbound Provisioning Endpoints” on page 538.

**Just-in-Time Provisioning**

At an SP site, PingFederate can also create and update local user accounts in an external LDAP directory or Microsoft SQL Server as part of SSO processing—**Just-in-Time (JIT) Provisioning**. This feature (formerly called Express Provisioning) allows SPs to maintain accounts for users who authenticate via IdP partners without having to provision accounts manually, when local accounts are required.

When configured, the PingFederate SP server writes user information to the local user store using attributes from the incoming SAML assertion. For SAML 2.0 partner connections, assertion attributes can be supplemented with user attributes returned from an Attribute Query (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).
PingFederate can also update existing user accounts based on assertions. When this option is enabled, PingFederate can add or overwrite attributes for a local user account each time SSO for a user is processed.

**Note:** Note that once user attributes are provisioned, they cannot be removed using JIT Provisioning. Where deprovisioning is required, we recommend using SCIM Inbound Provisioning.

For information about enabling JIT Provisioning, see “Choosing IdP Connection Options” on page 359. For configuration information, see “Using Just-in-Time Provisioning” on page 403.

## Federation Planning Checklist

An essential first step in establishing an identity federation involves discussions and agreements between you and your connection partners. The sections below comprise a partial checklist of items that should be coordinated before you deploy PingFederate.

**Tip:** Extensive coordination and configuration may be avoided by using Auto-Connect (see “Using Auto-Connect” on page 31).

**Standards and Specifications** Choose which federation protocol(s) your deployment will support. For SAML SSO configurations, decide which profiles and bindings will be used. (See the “Supported Standards” chapter in Getting Started.)

**Signing and Validation** Decide which SAML messages—assertions, responses, requests—will be digitally signed and how the messages will be verified by your federation partner. If messages are signed, decide how certificates will be exchanged (for example, secure email). (See “Security Infrastructure” on page 27.)

Also, if a stronger signature algorithm is required, determine what RSA algorithm will be used for signing. (The optional algorithm selection is available throughout the administrative console, where signing certificates are specified for various uses.)

**Back-Channel Security** Determine what type of SOAP channel authentication will be used: Basic or SSL/TLS. If SSL/TLS is used, determine whether server-only or both server and client certificates will be needed and how they will be managed. Also decide what level of security will be required for connections to back-end data stores or identity management systems.

**Trusted Certificate Management** Determine whether both partners are using SSL/TLS and/or signing certificates that have been signed by a major CA. (If self-signed certificates or nonstandard CAs are used, the signed certificates must be exchanged and imported into Trusted Certificate stores.) Also, determine whether you want to adopt a trust model that uses embedded certificates (see “Trust Models” on page 29).
Deployment  Decide how PingFederate fits into your existing network. Also, determine whether high-availability and/or failover options are required (see the PingFederate Server Clustering Guide).

Federation Server Identification  Determine how you and your partner(s) will identify your respective federation deployments. Under federation standards, both the sender (IdP) and the receiver (SP) of an assertion must be uniquely identified within the identity federation (see “Configuration Data Exchange” on page 41).

With PingFederate, you define a unique ID for each supported protocol (see “Specifying Federation Information” on page 109). Optionally, you can also use a list of multiple Virtual Server IDs on a connection-by-connection basis (see “Multiple Virtual Server IDs” on page 39).

Tip: PingFederate also provides for virtual host names, which differ from virtual server IDs (but are not mutually exclusive); they are intended to be used when your network configuration is such that you receive federation messages under more than one domain name (see “Using Virtual Host Names” on page 73).

Server Clock Synchronization  Ensure that both the SP and IdP server clocks are synchronized. SAML messages and STS tokens provide a time window that allows for small synchronization differentials. However, wide disparities will result in assertion or request time-outs.

User Data Stores  Identify the type of data store that contains user data when needed: LDAP, JDBC, or Custom (see “Data Stores” on page 23).

Web Application and Session Integration  Decide how PingFederate as an IdP receives subject identity information, either from an STS token or a user session.

For an SP, decide how PingFederate will forward user identity information to the destination Web application or system to start a session.

(See “SSO Integration Kits and Adapters” on page 14 and “Token Processors and Generators” on page 7.)

Transaction Logging  PingFederate provides basic transaction logging and monitoring. Decide whether transaction logging should be integrated with a systems management application and whether you have regulatory compliance requirements that affect your logging processes. (For more information, see “Managing Log Files” on page 44.)

Identity Mapping  For browser-based SSO, decide whether you will use PingFederate to link accounts on your respective systems using a persistent name identifier, or whether you will use account mapping (see “Identity Mapping” on page 17).

Attribute Contract Agreement  If your federation partnership will not use account linking, or will not use it exclusively, then you and your partner must agree on a set of attributes that the IdP will send in an assertion for either SSO or Web Service access. (For more information, see “Attribute Contracts” on page 20.)
Chapter 1
Key Concepts

**Metadata Exchange** If you are using SAML, decide whether you will use the metadata standard to exchange XML files containing configuration information. PingFederate makes it easy to use this protocol, which provides a significant shortcut to setting up your partner connections. (If your partner is also using PingFederate or supports standards permitting runtime metadata exchange, the process can be even simpler—see “Using Auto-Connect” on page 31.

**Multiple Virtual Server IDs**

Virtual server IDs provide more configuration flexibility in cases where you need to identify your server differently when connecting to a partner in one connection for multiple environments or in multiple connections where the partner also supports multiple federation IDs.

**Connecting to a Partner in One Connection**

This is a use case where you need to connect to multiple environments serviced by the same partner using one federation ID—multiplexing one SP connection to access multiple subdomain accounts in Microsoft Office 365.

Suppose both the marketing and the engineering departments of contoso.com (the IdP) have their own departmental subdomains, marketing.contoso.com and engineering.contoso.com. They are both registered in Office 365 (the SP) under the parent domain, contoso.com.

In this scenario, the PingFederate IdP server can be configured to include both marketing.contoso.com and engineering.contoso.com as the virtual server IDs in the Office 365 SP connection. Each virtual server ID has its own set of protocol endpoints, which can be obtained in the connection metadata (see “Exporting Metadata” on page 57).

After providing the protocol endpoints information to Office 365, when Office 365 sends login requests to PingFederate, PingFederate picks the correct IdP adapter to authenticate the end users based on the virtual server ID in the requests.

For each successful login, PingFederate builds an assertion with issuer being set to the corresponding virtual server ID. When Office 365 receives the assertion, it creates the end user session with the right subdomain settings based on the issuer value in the assertion.

**Connecting to a Partner in Multiple Connections**

In this use case, you connect to your partner in multiple connections. In each connection, you identify yourself and your partner differently.

For example, you as the SP provide separate environments for the end users based on their regions. Your IdP operates in two regions, Europe (EU) and North America (NA); their federation IDs are eu.idp.local and na.idp.local, respectively.
In the PingFederate SP server, you can create two IdP connections to federate identities for end users from both regions as follows:

<table>
<thead>
<tr>
<th>IdP connection #1</th>
<th>Partner's federation ID</th>
<th>Your virtual server ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>eu.idp.local</td>
<td>idp-eu.sp.tld</td>
</tr>
<tr>
<td>IdP connection #2</td>
<td>na.idp.local</td>
<td>idp-na.sp.tld</td>
</tr>
</tbody>
</table>

Based on the issuer (the partner’s federation ID) and the audience values (your virtual server ID), PingFederate determines at runtime which IdP connection the assertion is intended for, validates as per the connection settings, and passes attribute values to the SP adapter to create the end-user session.

**Working with Multiple Virtual Server IDs**

You can assign virtual server IDs either as an IdP during configuration of an SP connection (see “General Information” on page 255) or as an SP configuring an IdP connection (see “General Connection Information” on page 360) for both Browser SSO Profiles and WS-Trust STS (for access to identity-enabled Web Services).

If a connection has only one virtual server ID, it becomes the default virtual server ID for the connection. If the list contains several entries, you must specify one of them as the default virtual server ID for that connection. The default virtual server ID is used when no virtual server ID information is included in a request (see “IdP Endpoints” on page 529 as an IdP or “SP Endpoints” on page 533 as an SP).

In a connection with multiple virtual server IDs, you can optionally restrict each adapter added to the connection to certain virtual server IDs to enhance the end-user experience (see “Restricting an IdP Adapter to certain Virtual Server IDs” on page 269 and “Restricting an SP Adapter to certain virtual Server IDs” on page 373).

**Tip:** You can also restrict each token processor or token generator added to a WS-Trust STS SP connection or IdP connection, (see “Restricting a Token Processor to certain Virtual Server IDs” on page 458 or “Restricting a Token Generator to certain Virtual Server IDs” on page 484).

**Important:** To protect against unauthorized access, configure Issuance Criteria to verify virtual server ID in conjunction with other conditions, such as group membership information. For more information, see “Specifying Issuance Criteria (Optional)” on page 285 as an IdP or “Identifying Issuance Criteria (Optional)” on page 385 as an SP.

**Configuration Data Exchange**

If your partner’s deployment does not produce or consume a metadata file that conforms to SAML metadata specifications, you may need to exchange connection information manually. The following sections list some common
configuration details that must be exchanged if metadata files are not used. (These lists are not exhaustive.)

**IdP to SP**

If you are the IdP, your SP partner will need some or all of the following connection information (depending upon which profiles and bindings you are configuring):

- **Unique ID**—Identifies the IdP that issues an assertion or other SAML message. For SAML 2.0, the ID is the IdP Entity ID; for SAML 1.x, it is the IdP Issuer; for WS-Federation, it is the IdP Realm. PingFederate also supports the optional use of virtual IDs (see “Federation Server Identification” on page 38).
- **SOAP Artifact Resolution URL**—The endpoint your site uses to receive an SP’s SOAP requests when the artifact binding is used.
- **Single Logout Service URL**—The destination of SLO request messages.
- **Single Sign-On Service URL**—The endpoint where you receive and process assertions.

**SP to IdP**

If you are the SP, your IdP partner will need some or all of the following connection information (depending upon which profiles and bindings you are configuring):

- **Unique ID**—Identifies the SP. For SAML 2.0, the ID is the Entity ID; for SAML 1.x, it is the SP’s Audience; for WS-Federation, it is the SP’s Realm. PingFederate also supports the optional use of virtual IDs (see “Federation Server Identification” on page 38).
- **SOAP Artifact Resolution Service URL**—The endpoint to use for SOAP requests when the artifact binding is used.
- **Single Logout Service URL (SAML 2.0)**—The destination of SLO request messages.
- **Assertion Consumer Service URL**—The location where the SP receives assertions.
- **Target URLs**—The URLs for the protected resources that a user is trying to access.

**Mutual Settings Between Parties**

Many settings must be mutually set by the parties. This information might include such items as:

- **Attributes**—User information that will be sent in an assertion, if any (see “About Attributes” on page 19).
- **Signing certificates**—The SAML and WS-Federation protocols specify a number of conditions under which digital signatures are either required or optional (these conditions are built into the PingFederate connection-setup screens).
• SOAP connection type and authentication style—For SAML connections using the back channel (using the artifact binding, for example), HTTP Basic authentication, SSL client certificate authentication, digital signatures, or some combination of the three is required. You and your partner must exchange the necessary credentials, certificates, and/or signing keys.
This chapter describes general administrative functions for PingFederate, including:

- “Starting and Stopping PingFederate” on page 44
- “Managing Log Files” on page 44
- “Exporting Metadata” on page 57
- “Signing XML Files” on page 61
- “Using the Configuration Archive Utility” on page 62
- “Account Management” on page 64
- “Alternative Console Authentication” on page 68
- “Managing Email Configuration” on page 71
- “Using Virtual Host Names” on page 73
- “Changing Configuration Parameters” on page 74
- “Installing a New License Key” on page 77
- “Automating Configuration Migration” on page 78
- “Outbound Provisioning CLI” on page 83
- “Customizing User-Facing Screens” on page 87
- “Configuring Password Policy” on page 94
- “Extending the Lifetime of the PF Cookie” on page 94
- “Adding Custom HTTP Response Headers” on page 95

**Note:** The information in this chapter is presented from the viewpoint of an administrative user with “Admin” permissions (see “Account Management” on page 64).
Starting and Stopping PingFederate

(Windows)

To start PingFederate:

- From Start > Run dialog or a command prompt, run the batch file:
  ```
  <pf_install>\pingfederate\bin\run.bat
  ```
  Or:
  - Open the `\bin` folder in Windows and double-click the `run.bat` file.
  - Wait a moment for the script to execute. The server is started when you see the message “PingFederate started in [xx]:[yy]ms” in the command window, near the end of the start-up sequence.

To shut down PingFederate:

1. Enter Ctrl+C in the command-prompt window.
2. Enter y to terminate the batch script when prompted.

(Unix and Linux)

To start PingFederate:

1. From a command prompt, change directories to `<pf_install>/pingfederate/bin`.
2. Execute the `run.sh` file.
   - Wait a moment for the script to execute. The server is started when you see the message “PingFederate started in [xx]:[yy]ms” in the command window, near the end of the start-up sequence.

To shut down PingFederate:

- Enter Ctrl+C in the terminal window.

(All Platforms)

To access the PingFederate administrative console:

- Launch a Web browser and go this location:
  ```
  https://<DNS_NAME>:<port>/pingfederate/app
  ```
  where `<DNS_NAME>` is the fully qualified name of the machine running the PingFederate server and `<port>` is the port where the administrative console listens. The default port is 9999.

Managing Log Files

PingFederate generates these logs that document server events:

- `admin.log` — Records all actions performed by administrative-console users (see “Administrator Audit Logging” on page 46).
- `admin-api.log` — Records all actions performed by administrative-api users (see “Administrator API Audit Logging” on page 47).
- `transaction.log` — Records individual identity-federation runtime transactions at specified levels of detail.
  - The level of detail can be set globally or on a connection-by-connection basis (see “Runtime Transaction Logging” on page 47).
Managing Log Files

- audit.log — Records a selected, configurable subset of transaction log information plus additional details, intended for security-audit and regulatory compliance purposes (see “Security Audit Logging” on page 48).
- provisioner-audit.log — Records Outbound Provisioning events, intended for security-audit purpose (see “Outbound Provisioning Audit Logging” on page 50).
- server.log — Records all PingFederate runtime and administrative server operational activity (see “Server Logging” on page 51). PingFederate provides a utility for filtering server log entries (see “Using the Server Log Filter” on page 51).
- provisioner.log — Records only provisioning activity.
  (For more information, see “Outbound Provisioning for IdPs” on page 34.)
- init.log — Records only Jetty messages generated prior to PingFederate start up.

**Tip:** PingFederate logs user attributes, when they are present, in the server log, the transaction log, and the audit log (if configured). When privacy is required for sensitive user attributes, you can configure PingFederate to obfuscate (mask) their values in these logs (see “Attribute Masking” on page 23).

The logs are stored by default in the `<pf_install>/pingfederate/log` directory. You can change the location to any network directory by using the runtime parameter `pf.log.dir` (see “Changing Configuration Parameters” on page 74).

**Note:** The server, provisioner, and audit logs may be output to alternate formats, including database tables (see “Writing Logs to Other Formats” on page 53).

Other logs contained in the `pingfederate/log` directory are generated by the PingFederate Web container. These logs, `<date>.request[2].log`, record all HTTP requests for the given date. Properties controlling request logging are contained in the Web-container configuration files `jetty-runtime.xml` and `jetty-admin.xml` located in the PingFederate installation directory: `pingfederate/etc`.

The PingFederate-generated logs are controlled through the `log4j.xml` file located in `pingfederate/server/default/conf/`. See comments in the file for more information. Any changes to the log configuration requires a server restart.

**Note:** By default, initial server requests are always recorded in the server.log with a tracking ID number, which is then used to identify subsequent, related transactions. This ID can be useful for debugging and support purposes to aggregate and analyze log entries tied to the same original request. The ID may also be added to the configuration for transaction.log or audit.log, or it may be removed from the server.log `<layout>` element in the `log4j.xml` file, as needed for performance considerations.

Refer to the log4j open-source project for more information about logging levels and other configuration parameters (logging.apache.org/log4j/1.2/manual.html).
By default, PingFederate installs with a highly verbose level of logging. However, verbose logging may have a performance impact and clutter the log files. You may choose to lower the level, but we recommend that you not set it below WARN. For the transaction.log and the audit.log, note that any setting below INFO turns logging off.

Important: The transaction.log, the admin.log, the audit.log and the provisioner.log and the provisioner-audit.log files roll over at midnight each day. The system keeps all of the resulting historical log files. The transaction.log, audit.log and provisioner-audit.log files can become quite large, depending on your production load and settings; you may want to back up or remove older files on a routine basis.

Other PingFederate log files roll over when they reach 10MB. The system keeps five old log files of each type before overwriting the oldest. (This number can be changed in the log4j.xml file.)

The following sections provide more information about the primary logs:
- “Administrator Audit Logging” on page 47
- “Administrator API Audit Logging” on page 47
- “Runtime Transaction Logging” on page 47
- “Security Audit Logging” on page 48
- “Outbound Provisioning Audit Logging” on page 50
- “Server Logging” on page 51
- “Writing Logs to Other Formats” on page 53

Administrator Audit Logging

PingFederate records actions performed by server administrators. This information is recorded in the admin.log file. While the events themselves are not configurable, log4j.xml configuration settings may be adjusted to deliver the desired level of detail surrounding each event.

Each entry in the admin.log file is on a separate line and represents a single administrator action. The general format of each entry is the same, though specific events are recorded with information relevant to each type. Events are recorded when the corresponding Save button in the administrative console is clicked.

A log entry is generated for each of the events listed below:
- Password change or reset
- Account activation or deactivation
- Role change
- Login attempt
- Explicit user logouts (no time-outs)
- Data store created, modified, or deleted
- Certificate management
- SP connection created, modified, or deleted
- IdP connection created, modified, or deleted
- URL-to-adapter mapping management
Managing Log Files

- SP Adapter created, modified, or deleted
- IdP Adapter created, modified, or deleted
- Server settings management
- Metadata export
- Configuration export
- IdP Discovery management
- SP Affiliation created, modified, or deleted
- Attribute requester mapping
- IdP default URL modified
- SP default URLs modified
- XML file signatures applied

Each log entry contains information relating to the event, including:
- The time the event occurred on the PingFederate server.
- The username of the administrator performing the action.
- The role(s) assigned to the administrator at the time the event occurred.
- The type of event that occurred.
- Details about the event.

Each of the above fields is separated by a vertical pipe (|) for easier parsing.

Administrator API Audit Logging

PingFederate records actions performed via the administrative API (see “OAuth Access Grant Management Service” on page 566). This information is recorded in the `admin-api.log` file. While the events themselves are not configurable, you may adjust the `log4j.xml` configuration settings to alter the format and desired level of detail surrounding each event.

Each log entry contains information relating to the event, including:
- Time the event occurred on the PingFederate server
- Administrator username performing the action
- Authentication method
- Client IP
- HTTP method
- REST endpoint
- HTTP status code

Each of the above fields is separated by a vertical pipe (|) for ease of parsing.

Runtime Transaction Logging

PingFederate provides for flexible, scalable logging of all federated-identity transactions (inbound and outbound XML messaging). Transaction logging can be configured to any of four modes on a connection-by-connection basis (see “General Information” in either the IdP or SP “SSO Configuration” chapters).

You also have the option of overriding transaction logging for all connections (to find this feature, click the relevant Manage All... link under IdP/SP Connections on the Main Menu). You may want to use this override for
troubleshooting or as a one-step means of raising or lowering all connection logging
modes to the same level.

**Transaction Logging Modes**

The table below describes the four transaction logging modes:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No transaction logging.</td>
</tr>
</tbody>
</table>
| Standard  | Logs summary information for each transaction message, including:  
- Timestamp  
- Hostname:Port  
- Log Mode  
- Connection ID  
- SAML Status Code (for SAML responses only)  
- Context  
- Message Type  
- SAML ID (for SAML messages only)  
- Endpoint (for outbound messages only)  
- Target URL (if SSO transaction) |
| Enhanced  | Includes everything logged at the Standard level plus:  
- SAML_SUBJECT*  
- Binding  
- Relay State (if available)  
- Signature Policy  
- Signature Status  
- HTTP Request Parameters (outbound messages only)  
* Only when available in a SAML assertion, a single-signoff logout request, a Request Security Token Response (RSTR), or an authentication request (AuthnRequest) |
| Full      | Includes everything logged at the Enhanced level plus the complete XML message for every transaction. |

Each of the above fields is separated by a vertical pipe (|) for easier parsing.

**Security Audit Logging**

The PingFederate audit.log records a selected subset of transaction log information at runtime plus additional details, intended to facilitate security auditing and regulatory compliance. Elements of this log are described in the following table...
and configurable in the log4j.xml file located in pingfederate/server/default/conf.

**Note:** The audit log records SSO, SLO, OAuth AS, WS-Trust STS, and Inbound SCIM transactions. Outbound Provisioning transactions are not included; they are logged to the provisioner audit log (see “Outbound Provisioning Audit Logging” on page 50).

Audit-log elements may be output to different formats, including databases. For information, see “Writing Logs to Other Formats” on page 53.

**Table 4: Audit Log Configuration**

<table>
<thead>
<tr>
<th>Item*</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%d</td>
<td>Transaction time.</td>
</tr>
<tr>
<td>adapterid</td>
<td>The ID of an adapter instance.</td>
</tr>
<tr>
<td>app</td>
<td>The target SP application (when available).</td>
</tr>
<tr>
<td>assertionid</td>
<td>The unique ID for the SAML assertion.</td>
</tr>
<tr>
<td>attributes</td>
<td>User attributes received (for an SP log) or sent (for an IdP log).</td>
</tr>
<tr>
<td>connectionid</td>
<td>The connection identifier associated with the transaction.</td>
</tr>
<tr>
<td>description</td>
<td>Description of an authentication failure (when information is available from an IdP adapter).</td>
</tr>
<tr>
<td>event</td>
<td>The type of transaction (e.g. SSO).</td>
</tr>
<tr>
<td>granttype</td>
<td>OAuth grant type.</td>
</tr>
<tr>
<td>host</td>
<td>PingFederate host name or IP address.</td>
</tr>
<tr>
<td>initiator</td>
<td>(SAML 2.0 only) The federation role that initiated the SSO or SLO: SP or IDP.</td>
</tr>
<tr>
<td>inmessagetype</td>
<td>Incoming message type. Possible values are Request or Response.</td>
</tr>
<tr>
<td>inresponseto</td>
<td>The value of the InResponseTo attribute of an SSO or SLO Response.</td>
</tr>
<tr>
<td>inxmlmsg</td>
<td>The incoming XML message.</td>
</tr>
<tr>
<td>ip</td>
<td>Incoming IP address.</td>
</tr>
<tr>
<td>localuserid</td>
<td>The local ID used for the transaction (when account linking is enabled at the SP).</td>
</tr>
<tr>
<td>outxmlmsg</td>
<td>The outgoing XML message.</td>
</tr>
<tr>
<td>pfversion</td>
<td>The PingFederate version.</td>
</tr>
<tr>
<td>protocol</td>
<td>The associated identity protocol (e.g., SAML 2.0).</td>
</tr>
<tr>
<td>requestid</td>
<td>The ID of a request.</td>
</tr>
<tr>
<td>responseid</td>
<td>The ID of a response.</td>
</tr>
</tbody>
</table>
Outbound Provisioning Audit Logging

The PingFederate provisioner-audit.log records Outbound Provisioning transactions, intended to facilitate security auditing. Elements of this log are described in the following table and configurable in the log4j.xml file located in pingfederate/server/default/conf.

**Important:** Monitoring Outbound Provisioning transactions using JMX has been deprecated. By default, PingFederate logs outbound provisioning events to provisioner-audit.log (see “Runtime Monitoring Using JMX” on page 102). Elements from the provisioner audit log may be output to different formats, including databases. For information, see “Writing Logs to Other Formats” on page 53.

### Table 4: Audit Log Configuration (Continued)

<table>
<thead>
<tr>
<th>Item*</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>responsetime</td>
<td>Time elapsed (in milliseconds) from when a final request for a transaction is received to when the audit message is written. This value serves as an approximation of total transaction processing time and may be useful for monitoring trends.</td>
</tr>
<tr>
<td>role</td>
<td>The partner’s role in the transaction.</td>
</tr>
<tr>
<td>status</td>
<td>Transaction success or failure.</td>
</tr>
<tr>
<td>stspluginid</td>
<td>For STS transactions, the ID for the token-processor or token-generator instance.</td>
</tr>
<tr>
<td>subject</td>
<td>The subject of the transaction.</td>
</tr>
<tr>
<td>trackingid</td>
<td>The tracking ID used for debugging purposes in the server log (see the last “Note” on page 45 in the section “Managing Log Files”).</td>
</tr>
<tr>
<td>validatorid</td>
<td>The ID of the Password Credential Validator used for OAuth Resource Owner Grant transactions (see “Resource-Owner Credentials Mapping” on page 181).</td>
</tr>
</tbody>
</table>

* Italicized items are not configured by default for this log but may be added. Refer to log4j.xml for syntax requirements.

### Table 5: Provisioner Audit Log Configuration

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%d</td>
<td>Transaction time.</td>
</tr>
<tr>
<td>cycle_id</td>
<td>The unique ID for each provisioning cycle.</td>
</tr>
<tr>
<td>channel_id</td>
<td>The unique ID for Provisioning Channel between source and target.</td>
</tr>
</tbody>
</table>
Managing Log Files

Server Logging

The server log records all PingFederate runtime and administrative events, including status and error messages that can be used for troubleshooting. By default, the information is also sent to the terminal or command window running the PingFederate server.

Note: Alternatively, you can output the server log to a database (see “Writing Logs to Other Formats” on page 53).

To facilitate troubleshooting, administrators can use a filter utility to aggregate related events (see “Using the Server Log Filter” on page 51).

Elements of this log are described in the following table.

Table 5: Provisioner Audit Log Configuration (Continued)

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_type</td>
<td>The type of provisioning events, such as CREATE and UPDATE.</td>
</tr>
<tr>
<td>source_id</td>
<td>The provisioning Source ID.</td>
</tr>
<tr>
<td>target_id</td>
<td>The provisioning Target ID.</td>
</tr>
<tr>
<td>is_success</td>
<td>A flag to show whether the event was successful or not. If the attempt succeeded, the value is true; otherwise, the value is false.</td>
</tr>
<tr>
<td>non_success_cause</td>
<td>Description of failure cause.</td>
</tr>
</tbody>
</table>

Table 6: Server Log Components

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%d</td>
<td>Event date and time.</td>
</tr>
<tr>
<td>%X[trackingid]</td>
<td>The tracking ID for runtime events, used for debugging purposes (see the last “Note” on page 45 in the section “Managing Log Files”).</td>
</tr>
<tr>
<td>%p</td>
<td>Logging level.</td>
</tr>
<tr>
<td>%c</td>
<td>The Java class issuing the status or error message, when applicable.</td>
</tr>
<tr>
<td>%X[connectionid]</td>
<td>The partner ID associated with a runtime event.</td>
</tr>
<tr>
<td>%X[subject]</td>
<td>The SAML subject of the transaction, when applicable.</td>
</tr>
<tr>
<td>%m</td>
<td>Status or error message.</td>
</tr>
</tbody>
</table>

Using the Server Log Filter

PingFederate provides a utility that administrators can use to filter server logs. The tool, logfilter.bat|sh, is located in the <pf_install>/pingfederate/bin directory.
By default the utility sorts through all the server logs in the log directory, or you can move or copy one or more files to a different directory that can be specified as an input parameter.

The log filter returns lists of log entries based on either:
- Entity ID and Subject
- Tracking ID
- Session Cross-reference ID

The following table describes the utility’s command options. The table afterward describes optional parameters available for all of the commands.

**Table 7: Server Log Filter Command Options**

<table>
<thead>
<tr>
<th>Command Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>- entityid &lt;entity-id&gt;</td>
<td>These two commands must be used together and return a list of transactions for the specified federation partner’s entity ID and transaction subject.</td>
</tr>
<tr>
<td>- subject &lt;subject&gt;</td>
<td></td>
</tr>
<tr>
<td>- trackingid &lt;tracking_id&gt;</td>
<td>This command returns a list of transactions with the same tracking ID.</td>
</tr>
<tr>
<td>- sessionxrefid &lt;session_xref_id&gt;</td>
<td>This command returns a list of transactions for an ID assigned by PingFederate to associate different transactions according to the user session under which they occurred.</td>
</tr>
<tr>
<td></td>
<td>The value of &lt;session_xref_id&gt; may be the value of any of the following transaction tags in the target server log(s):</td>
</tr>
<tr>
<td></td>
<td>- Artifact</td>
</tr>
<tr>
<td></td>
<td>- Session Index</td>
</tr>
<tr>
<td></td>
<td>- Assertion ID</td>
</tr>
</tbody>
</table>

**Table 9: Server Log Filter Parameters**

<table>
<thead>
<tr>
<th>Parameter*</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>- logsdir &lt;log-files-directory&gt;</td>
<td>Full or relative path to source directory for the log(s).</td>
</tr>
<tr>
<td></td>
<td>Default: all server.log files in pf.log.dir</td>
</tr>
<tr>
<td></td>
<td>The installed location is:</td>
</tr>
<tr>
<td></td>
<td>&lt;pf_install&gt;/pingfederate/log</td>
</tr>
<tr>
<td></td>
<td>(For information about changing the default location, see “Changing Configuration Parameters” on page 74.)</td>
</tr>
</tbody>
</table>
Managing Log Files

The log filter creates its own log file, `logfilter.log`, located in the log directory. You can control settings for this log, as needed, in the file `logfilter.log4j.properties`, located in the `bin` directory.

Writing Logs to Other Formats

PingFederate provides the option of writing the audit, server, provisioner, and provisioner audit logs to commonly used databases (with failover to file logging).

For the audit log and the provisioner audit log, you may choose instead to write the information to the Common Event Format (see “Writing Audit or Provisioner Audit Logs to CEF” on page 55).

Finally, you may also configure PingFederate to write the audit log to a differently formatted log file that can be used by Splunk (see “Writing Audit Logs for Splunk” on page 56).

Writing Logs to Databases

You can enable database logging for the audit log, the server log, the provisioner log, and the provisioner audit log in the `log4j.xml` file in `pingfederate/server/default/conf`. Scripts for selected database types are provided to create the necessary table(s).

**Note:** Database logging replaces file logging. For the server log, database logging also replaces logging to the terminal or command window running PingFederate. Failover file logging is provided, however, in the event that database logging fails for any reason.

**Important:** Ensure that your database-driver JAR file is installed in the `pingfederate/server/default/lib` directory. You must restart the server after installing the driver.

---

**Table 10: Server Log Filter Parameters**

<table>
<thead>
<tr>
<th>Parameter*</th>
<th>Description</th>
</tr>
</thead>
</table>
| - outputfile `<output-file>` | Output path and file for the returned list.  
  Default: `pf.log.dir/logfilter_output.log`  
  The installed location is:  
  `<pf_install>/pingfederate/log`  
  (For information about changing the default location, see “Changing Configuration Parameters” on page 74.) |
| - outputtoconsole   | Returns list to the command console rather than to a file.                 |

*Optional for all commands.

The log filter creates its own log file, `logfilter.log`, located in the log directory. You can control settings for this log, as needed, in the file `logfilter.log4j.properties`, located in the `bin` directory.
To configure database logging:

1. In log4j.xml, uncomment one of the preset log-appender configurations listed below (or one from each list to configure all logs):
   - For the server log:
     - ServerLogToOracleDB (for Oracle)
     - ServerLogToSQLServerDB (for Microsoft SQL Server)
     - ServerLogToMySQLDB (for MySQL)
   - For the provisioner log:
     - ProvisionerLogToOracleDB (for Oracle)
     - ProvisionerLogToSQLServerDB (for Microsoft SQL Server)
     - ProvisionerLogToMySQLDB (for MySQL)
   - For the audit log:
     - SecurityAuditToOracleDB (for Oracle)
     - SecurityAuditToSQLServerDB (for Microsoft SQL Server)
     - SecurityAuditToMySQLDB (for MySQL)
   - For the provisioner audit log:
     - OutboundProvisionerEventToOracleDB (for Oracle)
     - OutboundProvisionerEventToSQLServerDB (for Microsoft SQL Server)
     - OutboundProvisionerEventToMySQLDB (for MySQL)

   **Note:** Each appender is followed by a related appender that creates a running *failover.log* file in the log directory. The failover appender must also be enabled (uncommented).

2. Replace placeholder parameter values for the appender(s).
   The parameter values provide access to the database. We recommend that they be tested and validated prior to production deployment.

   **Note:** See the NOTES in log4j.xml above the appender for more details.

   **Tip:** You can obfuscate the password used to access the database by running either `obfuscate.sh` or `obfuscate.bat`, located in `pingfederate/bin`. Use the actual password as an argument and copy the entire result into the value for the password parameter in log4j.xml.

3. Uncomment the appender reference in the associated logger element(s), as described in the appender NOTES:
   - For the server log:
     Uncomment the appender reference located under *Set up the Root Logger* near the end of the log4j.xml configuration file.
   - For the provisioner log:
     Uncomment the appender reference located under *Limit categories* near the end of the log4j.xml configuration file.
For the audit log:
Uncomment the appender in one or more of the following loggers located under Limit categories in the log4j.xml configuration file:
- org.sourceid.websso.profiles.sp.SpAuditLogger
- org.sourceid.websso.profiles.idp.IdpAuditLogger
- org.sourceid.websso.wstrust.log.STSAuditLogger
- org.sourceid.websso.profiles.idp.AsAuditLogger

**Note:** The last logger, idp.AsAuditLogger, enables database logging for interactions with the PingFederate OAuth Authorization Server.

For the provisioner audit log:
Uncomment the appender reference located under Limit categories near the end of the log4j.xml configuration file.

**Note:** As indicated in the IMPORTANT comment for the loggers, you must remove the existing appender reference(s).

4. (Optional) For the audit log and the provisioner audit log, you can configure elements for database logging in the ConversionPattern appender parameter, as needed (see Table 4 on page 49 for the audit log and Table 5 on page 50 for the provisioner audit log).

To create database tables:
- Scripts to create database tables for each of the three databases are provided for the audit, server log, and provisioner logs. The scripts are located in the directory:
  pingfederate/server/default/conf/log4j/sql-scripts

**Note:** The scripts are written to handle the default list of elements for the relevant database log-appender in log4j.xml. Any changes to the list requires corresponding changes to the SQL table-creation script (or to the table itself if it is already created).

Writing Audit or Provisioner Audit Logs to CEF

The Common Event Format (CEF) is an open logging standard. PingFederate provides an option of writing elements from audit log or provisioner audit log (or both) at runtime to a syslog receiver for parsing and analysis via HP ArcSight tools. Alternatively, you can write CEF to a flat file; however, using syslog, when available, is recommended.

You can enable this capability in the log4j.xml file (in pingfederate/server/default/conf).

**Note:** PingFederate is certified with HP ArcSight for interoperability using the default elements defined in log4j.xml. Any additions to these elements may render your CEF logging incompatible with HP ArcSight.

To configure CEF logging for the audit log:

1. In log4j.xml, uncomment one of the preset log-appender configurations:
   - SecurityAuditToCEFSyslog
Chapter 2
System Administration

- SecurityAuditToCEFFile

2. Replace the placeholder parameter value for the syslog host (if you are configuring the syslog appender).

3. Uncomment the chosen appender in one or more of the following loggers located under Limit categories in the log4j.xml configuration file:
   - org.sourceid.websso.profiles.sp.SpAuditLogger
   - org.sourceid.websso.profiles.idp.IdpAuditLogger
   - org.sourceid.wstrust.log.STSAuditLogger
   - org.sourceid.websso.profiles.idp.AsAuditLogger

   **Note:** The last logger, idp.AsAuditLogger, enables CEF logging for interactions with the PingFederate OAuth Authorization Server.

To configure CEF logging for the provisioner audit log:

1. In log4j.xml, uncomment one of the preset log-appender configurations:
   - OutboundProvisionerEventToCEFSyslog
   - OutboundProvisionerEventToCEFFile

2. Replace the placeholder parameter value for the syslog host (if you are configuring the syslog appender).

3. Uncomment the chosen appender for the ProvisionerAuditLogger logger located under Limit categories in the log4j.xml configuration file

   **Note:** As indicated in the IMPORTANT comment for the loggers, you must remove the existing appender reference(s) for syslog implementations.

Writing Audit Logs for Splunk

Splunk is enterprise software that allows for monitoring, reporting, and analyzing consolidated log files. Splunk captures and indexes real-time data into a single searchable repository from which reports, graphs, and other data visualization can be generated.

Ping Identity provides a custom Splunk App for PingFederate to process audit logs generated by a PingFederate deployment. The application provides rich system monitoring and reporting, including:

- Current transaction and system reports
- Service reports such as a daily usage report and IdP and SP reports per connection

   **Note:** OAuth AS and WS-Trust STS transactions are not currently supported for Splunk.

- Trend reports such as weekly and monthly usage reports, and trend analysis
The application uses a specially formatted version of the audit log (splunk-audit.log), which is written to the PingFederate log directory when the setup steps described below are followed.

**Note:** The Splunk App for PingFederate is available separately and requires enterprise-licensed (or trial) installation of the Splunk software and the Splunk Universal Forwarder, which is needed to collect data from the PingFederate audit log for Splunk. The application includes additional documentation on installation and available features. Download the free application from the Splunk Apps Web site “splunkbase” (splunk-base.splunk.com/apps)—search for PingFederate.

**To use the PingFederate Splunk App:**

1. Download and install the application in your Splunk environment (see **Note** above).

2. In your PingFederate installation, open the file log4j.xml in the directory: 
   `<pf_install>/pingfederate/server/default/conf`

3. In the log4j.xml file, depending on whether your PingFederate is operating as an SP or an IdP, search for either:
   
   logger name="org.sourceid.websso.profiles.sp.

   Or:

   logger name="org.sourceid.websso.profiles.idp.

4. Replace the `<appender-ref>` attribute value for the applicable logger with the value for Splunk indicated in the associated comment.

   For example, for an SP the modified initial XML tag for the logger would read:

   `<logger name="org.sourceid.websso.profiles.sp.SpAuditLogger" additivity="false">
   `<level value="INFO" />
   `<appender-ref ref="SecurityAudit2Splunk"/>

   **Note:** For auditing of adapter-to-adapter events, you must enable both the IdP and SP loggers.

5. Save the log4j.xml file and start or restart PingFederate.

6. Download and install the Splunk Universal Forwarder on the machine running PingFederate.

   Configure the Universal Forwarder to monitor the splunk-audit.log in the PingFederate directory `<pf_install>/pingfederate/log`.

   For detailed installation and configuration instructions, consult the Splunk documentation accompanying the Universal Forwarder.

**Exporting Metadata**

For SAML deployments PingFederate supports the export and import of metadata files, which federation partners can use to expedite their configuration. You can export metadata for any connection (or select certain nonconnection-specific metadata for export) via the Main Menu. You can import your partner's metadata file, when available, at the beginning of the connection-configuration process (see
“Managing IdP Connections” on page 353 or “Managing SP Connections” on page 246).

Tip: Connection-metadata export is also available from the Manage Connections screens, which can be reached from the Main Menu via the Manage All . . . link under either SP Connections or IdP Connections.

To reach the Metadata Export task:

> Click Metadata Export under Administrative Functions on the Main Menu.

To export connection metadata:

1. If your PingFederate server is configured to act as both an IdP and an SP, indicate which type of configuration you will export and click Next.
2. On the Metadata Mode screen, choose the option to “Use a connection . . .” and click Next.
   For more information, see “Choosing the Metadata Export Mode” on page 59.
3. On the Connection Metadata screen, select the connection from the drop-down menu. If the selected connection contains two or more virtual server IDs, choose the virtual server ID that you want to use during the export. Click Next.

   Note: The protocol endpoints in the connection metadata are specific to the selected virtual server ID. Re-export the connection metadata for your partners after updating your virtual server IDs (see “Multiple Virtual Server IDs” on page 38).

4. (Optional) On the Metadata Signing screen, select a certificate to use for signing the metadata XML file and click Next.

   Note: If you want to include the public-key information in the signed XML file, select the Key Info option.
   For more information, see “Signing Metadata” on page 60.
5. On the Export & Summary screen click the Export button, save the file, and then click Done.

To export selected metadata:

1. If your PingFederate server is configured to act as both an IdP and an SP, indicate which type of configuration you will export and click Next.
2. On the Metadata Mode screen, select the option to “Select information . . .” and click Next.
   For more information, see “Choosing the Metadata Export Mode” on page 59.
3. If you support more than one federation protocol, select the desired protocol on the Protocol screen and click Next.

4. Configure any or all of the remaining steps in the task (click Next to skip steps). For information see:
   - “Defining a Metadata Attribute Contract” on page 59
   - “Including a Signing Key” on page 60
   - “Signing Metadata” on page 60
   - “XML Encryption Certificates” on page 61

5. (Optional) On the Metadata Signing screen, select a certificate to use for signing the metadata XML file and click Next.

   ![Note] If you want to include the public-key information in the signed XML file, select the Key Info option.

   For more information, see “Signing Metadata” on page 60.

6. On the Export & Summary screen click the Export button, save the file, and then click Done.

Choosing the Metadata Export Mode

If you want to export metadata for a specific connection, keep that default selection on the Metadata Mode screen. Or you may choose instead to export metadata that is not bound to specific connection endpoints at your site.

Note: If the PingFederate secondary SSL port is configured and you want to use it for the SOAP channel, select the checkbox on the screen. If client-certificate authentication is configured for the SOAP channel, the secondary port is required and you must check this box. (For more information, see the description of the runtime property `pf.secondary.https.port` in the table under “Changing Configuration Parameters” on page 74.)

Defining a Metadata Attribute Contract

The Attribute Contract screen allows you to define the attribute contract you want to export in the metadata. For more information, see “Attribute Contracts” on page 19.

To reach this screen:

1. Click SAML Metadata Export under Administrative Functions on the Main Menu.

2. On the Metadata Mode screen, click the button for selecting the information manually and click Next.
To add an attribute:
1. Enter an attribute on the Attribute Contract screen and click **Add**.
2. Continue to add attributes as needed and click **Next**.

To edit an attribute name:
1. Click **Edit** and make your change.
2. Click **Update**.

To delete an attribute:
   ▶ Click **Delete**.

Including a Signing Key

In a metadata file you can manually include the public key for partners to use to verify the digital signature you will use to sign SAML messages. For more information, see “Digital Signing and Decryption Keys and Certificates” on page 209.

To export your public signature verification key:
   ▶ Select the certificate from the drop-down list and click **Next**.

Signing Metadata

Choose your organizational signing certificate on the Metadata Signing screen. The metadata XML file must be signed using a certificate trusted by your federation partner.

To specify a certificate:
1. Select the certificate from the drop-down list.
   
   If you have not yet created or imported your certificate into PingFederate, click **Manage Certificates** (see “Digital Signing and Decryption Keys and Certificates” on page 209).
2. (Optional) If you have agreed to send your public key with the SAML message, select the checkbox to include the certificate. To include the raw key in the signature as well, select the **Include the raw key ...** checkbox.

3. (Optional) Select the Signing Algorithm from the drop-down list. The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

**XML Encryption Certificates**

In your metadata file you can manually include the XML encryption key and certificate your partners can use to encrypt SAML messages.

To export an XML encryption key:

- Select the key from the drop-down list and click **Next**.

  If the certificate is not shown, click **Manage Certificates** to import it.

**Completing the Export**

On the Export & Summary screen, you can complete the XML-file download or change any information by clicking any of the headings in the Summary.

**Important:** To finish the download, you must click the **Export** button at the bottom left of the Export Metadata screen.

**Signing XML Files**

PingFederate supports digital signing of SAML metadata files or any other XML files that you and your partner might want to exchange. A signature applied to an XML file ensures that the file is from the original source and that its contents have not been modified by a third party.

When you configure a partner connection, you can also verify and import signed metadata files. For information:

- As an SP configuring an IdP connection, see “Importing IdP Metadata” on page 359.
- As an IdP configuring an SP connection, see “Importing Metadata” on page 254.
Chapter 2
System Administration

To sign an XML file
1. On the Select XML File screen, locate and open the file.
2. On the Digital Signature Settings screen, choose the certificate containing your signing key from the drop-down list.
3. On the Export & Summary screen, click the Export button to save the signed file.

**Note:** By default, certificate and public-key information is included in the signed XML file. If you do not wish to include this information, clear the KeyInfo and the KeyValue checkboxes.

**Important:** Be sure to click Export in the lower-left portion of the Export & Summary screen; clicking Done does not complete the operation.

Using the Configuration Archive Utility

PingFederate's archive utility allows you to export the current administrative-console configuration to a ZIP file and to import an existing archive for immediate deployment into a running PingFederate server.

**Tip:** A time-stamped configuration archive is created as a backup automatically every time you log on to the administrative console and before an existing archive is imported. The archives are stored in pingfederate/server/default/data/archive.

**Tip:** PingFederate includes a separate command-line utility that provides a finer level of control over configuration migration, as well as providing for scripting of routine configuration-management tasks (see “Automating Configuration Migration” on page 78).
Configuration archives can be used as backups or to transfer data from one PingFederate server to another, including migration to a new PingFederate release.

**Important:** Draft connections in archives are not imported. Complete any unfinished partner connections if you wish to include them in a full backup archive or in an archive to be used for configuration migration.

**Note:** The archive utility is intended for administrative-console configuration data only. It does not include error-page or other end-user HTML templates (see “Customizing User-Facing Screens” on page 87), nor any files under `<pf_install>/server/default/conf`. If any changes have been made to the default templates or configuration files, you must copy them over to new installations or other instances of PingFederate (assuming the changes are applicable).

The archive also does not capture license files, adapter JAR files, database drivers, or any other plug-ins; these also must be copied into any new instance of PingFederate. The import utility checks for and reports on any missing components (see “Using the Archive Import Screen,” next).

**To reach this screen:**

- Click **Configuration Archive** on the Main Menu.

**To create an archive:**

1. On the Select Import/Export screen, ensure Export is selected and click **Next**.
2. On the Export screen, click **Export**, save the download to your file system, and click **Done**.

**To import and deploy an archive:**

- On the Select Import/Export screen, click **Import** and then **Next**.

  See the next section, “Using the Archive Import Screen,” for more information.

**Note:** Alternatively, you can deploy an archive manually, using the procedure below.

**Caution:** Deploying a configuration archive, either manually or by using the administrative console, always overwrites all existing configuration data.

**To deploy an archive manually:**

1. Copy a previously stored archive into the directory:
   `<pf_install>/pingfederate/server/default/data/drop-in-deployer`
2. Rename the copied file to `data.zip`.

   When the PingFederate server is running, the file is again renamed with a timestamp after a moment and the data automatically deploys, replacing the current UI configuration. Restarting the server is not required.
3. Consult the PingFederate server start-up window, or the server log file, for any messages concerning missing plug-in components or other errors.

**Note:** A data archive imported via the drop-in-deployer directory is deployed by default regardless of errors, unlike the default behavior using the Import screen.

### Using the Archive Import Screen

When you initiate deployment of a configuration archive using the Import screen, PingFederate displays error messages if there are any missing plug-in components (such as adapters, database drivers, or token translators) on which the archive depends, or any mismatches of PingFederate licensing authorization.

If there are missing components or license inconsistencies, the import is halted by default to allow you to install the necessary components or license. However, you can choose to force the deployment and then install the necessary files later.

**Note:** Installation of any missing database drivers or other third-party libraries will require a PingFederate server restart.

**To deploy a configuration archive:**

1. On the Import screen, click **Browse** to locate the required ZIP file.
2. (Optional) Select **Force Import** to deploy the archive regardless of whether dependencies are detected.

**Important:** If you make this selection, consult the server start-up window or the server log for any errors.

3. Click **Import**.
4. Read the on-screen caution statement and indicate whether you want to continue.
   - When you click **Yes**, a message is displayed indicating the deployment result.
5. Click **Done**.
Account Management

PingFederate provides a choice of single- or multi-user system administration (see “Setting Administration Options” on page 98).

**Note:** This choice is not presented in the administrative console if you are using your network’s LDAP user store, a RADIUS server, or client certificates for authentication to the PingFederate administrative console (see “Alternative Console Authentication” on page 68).

If you choose native multi-user administration or if you are using alternative authentication, PingFederate provides role-based access control, as shown in the following table. For native multi-user administration, you can choose to use email for password setting and resetting notifications.

<table>
<thead>
<tr>
<th>Role Assignment</th>
<th>Access Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Admin</td>
<td>Create users, deactivate users, change or reset passwords, and install replacement license keys. (This role is not provided if you are using LDAP or RADIUS authentication for administrative logon, since user management is handled outside of PingFederate.)</td>
</tr>
<tr>
<td>Admin</td>
<td>Configure partner connections and most system settings (except user management and local key/certificate handling).</td>
</tr>
<tr>
<td>Crypto Admin</td>
<td>Manage local keys and certificates.</td>
</tr>
<tr>
<td>Auditor</td>
<td>View-only permissions for all administrative functions.</td>
</tr>
</tbody>
</table>

When Auditor is assigned, no other roles may be set. Admin users may have multiple roles set.

**Tip:** The same user may log on from more than one browser or location. Also, by default, more than one user can log on to PingFederate at a time. You can change this default to restrict the administrative console to one administrative user at a time (see “Changing Configuration Parameters” on page 74).

Any number of auditors may log on at any time, regardless of the property setting.

**Note:** For security, after three failed logon attempts from the same location within a short time period, the administrative console will temporarily lock out further attempts by the same user. The user must wait one minute to try again.

To reach the Account Management screen:

- Click **Account Management** under Administrative Functions on the Main Menu.
Note: If you are using alternative PingFederate authentication, the **Account Management** configuration is not available. Set PingFederate-specific permissions for users in configuration files located in `<pf_install>/pingfederate/bin` (see “Alternative Console Authentication” on page 68).

<table>
<thead>
<tr>
<th>USERNAME</th>
<th>USER ADMIN</th>
<th>ADMIN</th>
<th>CRYPTO ADMIN</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator</td>
<td>☑ Auditor</td>
<td>☑</td>
<td>☑</td>
<td>Deactivate / Change Password</td>
</tr>
<tr>
<td></td>
<td>☑ Admin</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Users with User Admin permissions can add other users, assign them any role, or reset their passwords, as well as change their own passwords. Other types of users can change only their own passwords from this screen (see “Changing Passwords” on page 68).

**To add a user:**

1. Click **Create User**.
2. On the User Information screen, enter the required fields (indicated by asterisks).
   
   **Note:** Only Username is required, unless you elected on the Account Management screen to have PingFederate send passwords via email, in which case you must supply an email address.

3. (Optional) Enter additional information.
4. Click **Next** to set up a password (see “Setting or Resetting Passwords” on page 67).

   **Important:** After you set the password and return to the Account Management screen, you must select permissions for the new user and click **Save** to complete the process.

**To define a user’s permissions:**

- Select or clear the checkboxes under the permission categories you want to assign or remove (see the table above).
  
  Clicking the **Auditor** button deactivates other permission selections.

**Note:** For traceability and accountability purposes, users cannot be deleted; their records are retained and they can be reactivated if needed.
To enable password notification:

1. Select the password-notification checkbox.
2. If you have not yet configured PingFederate to use your email server, click Email Server Settings and complete the configuration (see “Managing Email Configuration” on page 71).
3. Click Save (or Next if you are installing PingFederate).

**Note:** If you are setting up email notifications for the first time, you must click Email Server Settings and configure the settings. If you are not sure of the correct settings, enter placeholders on the Email Notification screen; you can return later and update the information.

Setting or Resetting Passwords

A user administrator can generate or assign temporary passwords for other users, either during user setup or at a later time (for example, if a user forgets his or her password).

**Note:** If you are using non-native authentication for PingFederate, password management is handled at the network level (see “Alternative Console Authentication” on page 68).

Initial or reset passwords may be used only once; the administrative console requires the user to change the password immediately after logging on.

To reach this screen:

1. Click Account Management on the Main Menu.
2. Click Reset Password under Action for a user.

To set or reset a user’s password:

1. Either:
   - Click Generate one-time password.
   Or:
   - Enter a temporary password in the text box. Password policy applies. For more information, see “Configuring Password Policy” on page 94.
2. Click Done.

**Important:** The password and any other changes, including new user records, are not stored until you click Save on the Account Management screen.
Chapter 2
System Administration

After you click **Save** on the Account Management screen, the new password is emailed to the user automatically, if you have enabled email notifications (see “Account Management” on page 64).

## Changing Passwords

Any user can change his or her own password. For information about resetting another user’s password (if you are a user administrator), see the previous section.

**Note:** If you logged on to PingFederate using your network ID and password, you can change your password only at the network level. The new password will apply to PingFederate automatically the next time you log on.

To change your password:

1. Click **Account Management** on the Main Menu.
2. On the Account Management screen, click **Change Password** under the Action column.
3. Enter your **Current Password** and **New Password** (plus confirmation) and click **Save**.

Password policy applies. For more information, see “Configuring Password Policy” on page 94.

**Important:** If you are the sole user administrator, take steps to ensure that you do not forget your new password.

## Alternative Console Authentication

As an alternative to using PingFederate’s own internal data store for authentication to the administrative console, you can configure PingFederate to use either your network’s LDAP user-data store, the RADIUS protocol, or client certificates. You can configure any of these alternatives at any time.

Note that most user-management functions are handled outside the scope of the PingFederate administrative console when alternative authentication is enabled.
Authorization levels, however, as described in “Account Management” on page 64, must be assigned in PingFederate configuration files.

**Tip:** You can use the LDAP configuration file in conjunction with RADIUS authentication to determine permissions dynamically via LDAP group assignments.

### Using LDAP Authentication

The LDAP authentication setup is available via configuration files in the `<pf_install>/pingfederate/bin` directory.

**Note:** When LDAP authentication is configured, PingFederate does not lock out administrative users based upon the number of failed logon attempts. Responsibility for preventing access is instead delegated to the LDAP server and enforced according to its password lockout settings.

To configure PingFederate to use network LDAP authentication:

1. In the `pingfederate/bin/run.properties` file, change the value of `pf.console.authentication` as shown below:
   ```properties
   pf.console.authentication=LDAP
   ```

   **Note:** You can restore internal user-management control at any time by returning the value to `native` and restarting the PingFederate server.

2. In the `pingfederate/bin/ldap.properties` file, change property values as needed for your network configuration.
   See the comments in the file for instructions and additional information.

   **Important:** Be sure to assign LDAP users or designated LDAP groups (or both) to at least one of the PingFederate administrative roles as indicated in the properties file. For information about permissions attached to the PingFederate roles, see “Account Management” on page 64.

   **Tip:** You can also use this configuration file in conjunction with RADIUS authentication to determine permissions dynamically via an LDAP connection.

3. Start or restart PingFederate.

### Using RADIUS Authentication

The RADIUS authentication setup is available via configuration files in the `<pf_install>/pingfederate/bin` directory. The RADIUS protocol provides a common approach for implementing strong authentication in a client-
server configuration. PingFederate supports the protocol scenarios for one- and two-step (for example, challenge-response) authentication.

**Note:** When RADIUS authentication is configured, PingFederate does not lock out administrative users based upon the number of failed logon attempts. Responsibility for preventing access is instead delegated to the RADIUS server and enforced according to its password lockout settings.

**Note:** The NAS-IP-Address attribute is added to all Access-Request packets sent to the RADIUS server. The value is copied from the pf.engine.bind.address property in run.properties. Only IPv4 addresses are supported.

To configure PingFederate to use network RADIUS authentication:

1. In the pingfederate/bin/run.properties file, change the value of pf.console.authentication as shown below:
   
   ```
   pf.console.authentication=RADIUS
   ```
   
   **Note:** You can restore internal user-management control at any time by returning the value to native and restarting the PingFederate server.

2. In the pingfederate/bin/radius.properties file, change property values as needed for your network configuration.
   
   See the comments in the file for instructions and additional information.

   **Important:** Be sure to assign RADIUS users or designated RADIUS groups (or both) to at least one of the PingFederate administrative roles as indicated in the properties file. Alternatively, you can set the use.ldap.roles property to true and use the LDAP properties file (also in the bin directory) to map LDAP group-based permissions to PingFederate roles. (For information about permissions attached to the PingFederate roles, see “Account Management” on page 64.)

3. Start or restart PingFederate.

**Using Certificate-Based Authentication**

To enable client-certificate authentication, PingFederate administrative users must have imported into their Web browsers an X.509 key and certificate suitable for user authentication. In addition, the corresponding root CA certificate(s) must be contained in the Java runtime or the PingFederate trusted store (see “Trusted Certificate Authorities” on page 204).

Other setup steps, including designating user permissions, are required via configuration files in the <pf_install>/pingfederate/bin directory.

To enable certificate-based PingFederate console authentication:

1. If not already done, import the necessary client key and certificate into the Web browser used to access PingFederate.
   
   Refer to the browser's documentation, as needed, for instructions.
2. Log on normally (using username and password) to the PingFederate console as a user with permissions that include the role Crypto Admin.
   The default administrator includes this role. (For more information, see “Account Management” on page 64.)

3. Ensure the client-certificate’s root CA and any intermediate CA certificates are contained in the trusted store (either for the Java runtime or PingFederate, or both).
   To import a certificate, click Trusted CAs in the Security section under Server Configuration.

   **Tip:** You may wish to click the Serial number and copy the Issuer DN to use at Step 5.

4. In the pingfederate/bin/run.properties file, change the value of pf.console.authentication as shown below:
   pf.console.authentication=cert

5. In the pingfederate/bin/cert_auth.properties file, enter the Issuer DN for the client certificate as a value for the property:
   rootca.issuer.x
   where \( x \) is a sequential number starting at 1 (see the properties file for more information).

   **Important:** The configuration values are case-sensitive.

If you copied the Issuer DN at Step 3, paste this value.

6. Repeat the previous step for any additional CAs as needed.

7. Enter the certificate user’s Subject DN for the applicable PingFederate permission role(s), as described in the properties file.

   **Important:** The configuration values are case-sensitive.

   See “Account Management” on page 64 for more information about PingFederate permissions.

8. Repeat the previous step for all users as needed.

   **Note:** Other settings in the properties file are used to display the user’s ID (the Subject DN) in abbreviated form in the administrative console.

9. Restart the PingFederate server (see “Starting and Stopping PingFederate” on page 44).

---

**Managing Email Configuration**

If you are using email notification for password resets, licensing events, or certificate-expiration warnings, you must set up and maintain a connection to the email server that PingFederate will use to send messages (see “Account Management” on page 64 and “Configuring Runtime Notifications” on page 99).
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>“From” Address</td>
<td>The email address that appears in the “From” header line in email messages generated by PingFederate. The address must be in valid format but need not be set up on your system.</td>
</tr>
<tr>
<td>Email Server</td>
<td>The IP address or hostname of your email server.</td>
</tr>
<tr>
<td>SMTP Port</td>
<td>The SMTP port on your email server (default: 25).</td>
</tr>
<tr>
<td>SSL SMTP Port</td>
<td>The secure SMTP port on your email server (default: 465). This field is not active unless Use SSL is enabled below.</td>
</tr>
<tr>
<td>Connection Timeout</td>
<td>The amount of time in seconds that PingFederate will wait before it times out connecting to the SMTP server.</td>
</tr>
<tr>
<td>Use SSL</td>
<td>(Optional) Requires the use of the SMTP secure channel.</td>
</tr>
<tr>
<td>Use TLS</td>
<td>(Optional) Requires the use of the secure transport layer.</td>
</tr>
<tr>
<td>Enable SMTP Debugging Messages</td>
<td>(Optional) Turns on detailed error messages for the PingFederate server log to help troubleshoot any problems,</td>
</tr>
<tr>
<td>Username</td>
<td>(Optional) Authorized email username.</td>
</tr>
</tbody>
</table>
Using Virtual Host Names

To configure access to your email server:
1. Enter information into required fields, at minimum (Username and Password are not required.)
2. (Recommended) Enter an email address (or addresses) in the Test Address field and click Test Email Connectivity.
   - A message next to the button indicates a successful test. Verify that the test email address received a message from the server.
   - Test reports are also written to the server.log file in the /log directory.

Using Virtual Host Names

In certain contexts, the SAML specifications require that XML messages include a URL identifying the host name to which the sender directed the message. (The name of the XML element containing the URL varies among protocols.) In addition, the recipient must verify that the value matches the location where the message is received.

Depending on your networking requirements, this specification can present problems—for example, in the case of proxy forwarding, where the final destination host name might be unknown to your federation partner. To provide more flexibility in such cases, you can set up a list of alternative host names for PingFederate to use as part of its message-security validation.

Note that virtual host names are used for a different purpose than virtual server IDs, which provide separate unique identifiers for a federation deployment, normally in the same domain (see “Federation Server Identification” on page 37). Depending on your needs, however, you can configure virtual server IDs and virtual hosts in the same installation of PingFederate.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>(Optional) User password.</td>
</tr>
<tr>
<td>Confirm Password</td>
<td>Re-entered password, when a Password is used.</td>
</tr>
<tr>
<td>Test Address</td>
<td>(Optional) An email address to use in conjunction with testing the connection.</td>
</tr>
</tbody>
</table>

To reach this screen:
- Click Email Configuration on the Main Menu under Administrative Functions.

If this link is not displayed, then no email notifications are configured (see “Configuring Runtime Notifications” on page 99 or “Account Management” on page 64).
Changing Configuration Parameters

PingFederate’s default administrative-console and runtime behavior is controlled in part by configuration properties contained in the file `run.properties`, located in: `<pf_install>/pingfederate/bin`. The table below describes the properties; refer to the file itself for default settings not specified here.

You can change these settings as needed. Restart the PingFederate server for changes to take effect.

**Note:** Properties related to server clustering and provisioning failover are described in the PingFederate Server Clustering Guide.

**Important:** If PingFederate is deployed in a cluster, changes to default settings for runtime-server properties must be applied to other server nodes manually. The settings in `run.properties` are not replicated using automated synchronization methods.

### Table 12: PingFederate Configuration Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pf.admin.https.port</td>
<td>Defines the port on which the PingFederate administrative console runs. Default is 9999.</td>
</tr>
<tr>
<td>pf.console.bind.address</td>
<td>Defines the IP address over which the PingFederate administrative console communicates. Use for deployments where multiple network interfaces are installed on the machine running PingFederate.</td>
</tr>
<tr>
<td>pf.console.title</td>
<td>Defines the browser window or tab title for the administrative console, used to make separate instances identifiable.</td>
</tr>
<tr>
<td>pf.console.session.timeout</td>
<td>Defines the length of time in minutes until an inactive administrative console times out. The minimum setting is 1 minute; maximum is 8 hours (480 minutes). Default is 30 minutes.</td>
</tr>
<tr>
<td>pf.console.login.mode</td>
<td>Indicates whether more than one Admin user may access the administrative console at one time (see “Setting Administration Options” on page 98). Values: Single</td>
</tr>
<tr>
<td>pf.console.authentication</td>
<td>Indicates whether administrators log on to PingFederate using credentials managed internally, by PingFederate, or externally (see “Alternative Console Authentication” on page 68).</td>
</tr>
</tbody>
</table>
### Changing Configuration Parameters

Table 13: PingFederate Configuration Properties (Continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldap.properties.file</td>
<td>When LDAP administrative-console authentication is enabled, indicates the</td>
</tr>
<tr>
<td></td>
<td>name of the file containing configuration properties.</td>
</tr>
<tr>
<td>cert.properties.file</td>
<td>When certificate-based console authentication is enabled, indicates the</td>
</tr>
<tr>
<td></td>
<td>name of the file containing configuration properties.</td>
</tr>
<tr>
<td>radius.properties.file</td>
<td>When RADIUS-based console authentication is enabled, indicates the</td>
</tr>
<tr>
<td></td>
<td>name of the file containing configuration properties.</td>
</tr>
<tr>
<td>pf.http.port</td>
<td>Defines the port on which PingFederate listens for unencrypted HTTP traffic</td>
</tr>
<tr>
<td></td>
<td>at runtime. For security reasons, this port is turned off by default.</td>
</tr>
<tr>
<td></td>
<td><strong>Caution:</strong> This port should remain disabled in production if your</td>
</tr>
<tr>
<td></td>
<td>deployment configuration directly exposes the PingFederate server to the</td>
</tr>
<tr>
<td></td>
<td>Internet.</td>
</tr>
<tr>
<td>pf.https.port</td>
<td>Defines the port on which PingFederate listens for encrypted HTTPS (SSL/TLS)</td>
</tr>
<tr>
<td></td>
<td>traffic. Default is 9031.</td>
</tr>
<tr>
<td>pf.secondary.https.port</td>
<td>Defines a secondary HTTPS port that can be used, for example, with SOAP</td>
</tr>
<tr>
<td></td>
<td>or artifact SAML bindings or for WS-Trust STS calls. To use this port,</td>
</tr>
<tr>
<td></td>
<td>change the placeholder value to the port number you want to use.</td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> If you are using mutual SSL/TLS for either WS-Trust STS</td>
</tr>
<tr>
<td></td>
<td>authentication or for SAML back-channel authentication, you must use this</td>
</tr>
<tr>
<td></td>
<td>port for security reasons (or use a similarly configured new listener, with</td>
</tr>
<tr>
<td></td>
<td>either WantClientAuth or NeedClientAuth set to true—see &quot;Note&quot; at the end</td>
</tr>
<tr>
<td></td>
<td>of this table).</td>
</tr>
<tr>
<td>pf.engine.bind.address</td>
<td>Defines the IP address over which the PingFederate server communicates with</td>
</tr>
<tr>
<td></td>
<td>partner federation gateways. Use for deployments where multiple network</td>
</tr>
<tr>
<td></td>
<td>interfaces are installed on the machine running PingFederate.</td>
</tr>
</tbody>
</table>
Table 13: PingFederate Configuration Properties (Continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pf.monitor.bind.address</td>
<td>Defines the IP address over which an SNMP agent and JMX communicate with PingFederate (see “Configuring Runtime Reporting” on page 100). Use for deployments where multiple network interfaces are installed on the machine running PingFederate.</td>
</tr>
<tr>
<td>pf.engine.prefer_ipv4</td>
<td>Defines the protocol to be used by PingFederate. True enables use of ipv_4 only. False (the default) enables use of both ipv_4 and ipv_6.</td>
</tr>
<tr>
<td>pf.runtime.context.path</td>
<td>Allows customization of the server path for PingFederate endpoints (see “Application Endpoints” on page 529). [Note: If this property is changed, the path must also be added to the Base URL for your server (see “Specifying Federation Information” on page 109).]</td>
</tr>
<tr>
<td>pf.log.dir</td>
<td>Network path to the output location of log files. The default is: &lt;pf_install&gt;/pingfederate/log</td>
</tr>
<tr>
<td>pf.hsm.mode</td>
<td>Enables or disables (the default) a FIPS-compliance Hardware Security Module (see Appendix A in Getting Started).</td>
</tr>
<tr>
<td>pf.provisioner.mode</td>
<td>Enables or disables (the default) Outbound Provisioning (see “Outbound Provisioning for IdPs” on page 34). Also used to enable provisioning failover (see the PingFederate Server Clustering Guide).</td>
</tr>
</tbody>
</table>

**Note:** Additional configuration of the listener ports (including adding new listeners) is available via the <pf_install>/pingfederate/etc/jetty-runtime.xml file. For example, options include WantClientAuth and NeedClientAuth flags, which indicate that a client certificate is either requested or required, respectively, for mutual SSL/TLS. (For the preconfigured SSL secondary port, WantClientAuth is set to true by default; NeedClientAuth is set to false.)
Installing a New License Key

If your license key is going to expire or has expired recently, you can install a new one without interrupting services by using the PingFederate administrative console.

Tip: You can configure the server to send administrators email warnings regarding the license status (see “Configuring Runtime Notifications” on page 99).

You also need to install a new license key for a new PingFederate installation as well as software upgrades (other than patch releases).

To request a new license key:
- Go to the Ping Identity website licensing page (www.pingidentity.com/support/licensing).

After you receive the new license key, install it using the administrative console, regardless of whether a previous license is installed or not.

Tip: When you use the administrative console and a previous license exists, PingFederate compares the new license with the existing one and warns of any potential problems (or aborts the import if the license is invalid for any reason). After the license key is imported, the previous license file is saved with a timestamp in the filename.

To install a license key using the administrative console (when no previous license is configured):

1. Start PingFederate and access the administrative console (see “Starting and Stopping PingFederate” on page 44).

2. On the Import License screen, click Browse to select the file, then click Import.

   The license file is verified for use with the current instance of PingFederate and copied as pingfederate.lic to:
   
   <pf_install>/pingfederate/server/default/conf
   
   This is referred as the primary license file.

   For a clustered-server environment, the running engine nodes consume the new license and save the information to <pf_install>/pingfederate/server/default/data/.pingfederate.lic. This is referred to as the secondary license file.

   Note: The secondary license file is used only when the primary license file cannot be found.

For any engine node that is stopped when a new license is imported through the administrative console, the new license is utilized when PingFederate restarts, and the new license information is saved to the secondary license file.

(For more information about clustering, see the Server Clustering Guide.)
To install a replacement license key using the administrative console:

1. Start PingFederate and access the administrative console.
2. Click License Management under Administrative Functions.
3. On the License Management screen, click Choose File to select the file and then click Import.

No filename restrictions are imposed for importing the file. It can be renamed before installation if necessary.

Once you import the new license file, the previous pingfederate.lic license file is renamed with a timestamp in the /conf directory:

If the new license does not include support for features covered by your existing license, or if there is some other potential problem with the license, you are advised and prompted on whether to continue.

---

**Note:** If the license is for the wrong version of PingFederate or is found to be invalid for some other reason, PingFederate displays the error(s) and reverts to the previous license.

For a clustered-server environment, the running engine nodes consume the new license and save the information to <pf_install>/pingfederate/server/default/data/.pingfederate.lic. This is referred to as the secondary license file.

---

**Note:** The secondary license file is used only when the primary license file cannot be found.

For any engine node that is stopped when a new license is imported through the administrative console, the new license is utilized when PingFederate restarts, and the new license information is saved to the secondary license file.

(For information about clustering, see the Server Clustering Guide.)

### Automating Configuration Migration

PingFederate provides a configuration-migration tool that can be used for scripting the transfer of administrative-console configurations and configuration property files from one PingFederate server to another—for example, from a test environment to production. The tool may also be used to manage certificates for the target server.

The command-line utility, configcopy in <pf_install>/pingfederate/bin, uses PingFederate's built-in Connection Management Web Service in conjunction with an internal Web Service to export and import connections and other configurations, and to obtain lists (see “Connection Management Service” on page 554).

---

**Important:** The Connection Management Service must be activated for both the source and target servers before the configcopy tool can be used (see “Authentication” on page 215.

---

**Caution:** For security reasons, the Management Service should be disabled whenever it is not in use.
Automating Configuration Migration

Copying the Key from the Source to the Target Server

You must copy the key from the source server to the target server before you migrate data using the configcopy tool. To do this, you copy the key (or keys, if you have more than one) from the `pf.jwk` file on the source server and append it to the last key in the `pf.jwk` file on the target server, and then restart that target server. This step only needs to be done before the first migration.

To copy the key from the source to the target servers:

1. In your PingFederate installation on the source server, open the `pf.jwk` file in the directory:
   `<pf_install>/pingfederate/server/default/data`
2. Copy the key in the file. Ensure you copy the entire key JSON message. For example, you would copy all the bolded text as shown below:
   
   ```json
   {"keys":[{"kty":"oct","kid":"j0PUEdAb95","k":"AGi8Lg_ewdx1-_30Cx83kDMQE9oNlhgJSa_Pc4I8J7U8"}]}
   ```
3. In your PingFederate installation on the target server, open the `pf.jwk` file in the directory:
   `<pf_install>/pingfederate/server/default/data`
4. Insert a comma at the end of the last key in the file and append the source key as shown below:
   
   ```json
   {"keys":[{"kty":"oct","kid":"wER9zEpaPe","k":"i0HQr9JmsqjAX4o_BQU1qGJzoLQ1-\nrmwp8u3GyHzTB8"},{"kty":"oct","kid":"j0PUEdAb95","k":"AGi8Lg_ewdx1-_30Cx83kDMQE9oNlhgJSa_Pc4I8J7U8"}]}
   ```
5. Save the `pf.jwk` file and start or restart the target server.
6. If applicable, repeat the steps above for each target server running PingFederate.

Administrative Console Migration

For migrating data configured with the source server's administrative console, this tool performs these overall processing steps:

1. Retrieves specified connection and/or other configuration data (XML) from a source PingFederate server.
2. Modifies the configuration with any changes required for the target environment, according to settings in one or more properties files and/or command-line arguments.
3. Imports the updated configuration into the PingFederate target server.

The configcopy tool can perform these functions in real time, from server to server, or by using an intermediate file. The latter option is useful when both the source and target PingFederate servers are either not running at the same time or not accessible from the same operating-system command window.

---

**Important**: For one-time configuration transfers from one version of PingFederate to a newer version, we recommend using a complete configuration archive, either with configcopy archive export/import commands or manually (see “Using the Configuration Archive Utility” on page 62). Other configcopy commands are not supported for this purpose.
Operational capabilities include:

- Listing of source partner connections, adapter or STS token-translator instances, outbound-provisioning channels, or data-store connections.
  
  List commands include optional filter settings, when applicable.
- Copying one or more partner connections, outbound-provisioning channels, or instances of adapters or token translators.
- Copying one or more data-store connections.
- Copying server settings.
- Exporting and importing full configuration archives.

### Configuration File Copying

The configcopy tool supports copying configuration files containing runtime properties (including those needed for server clustering) that may have been manually customized for the source configuration and need to be migrated. The file-copy command may also be used to copy the PingFederate internal, hypersonic database when needed.

### Certificate Management

Administrators may use the configcopy tool to perform the following certificate-management tasks on the target PingFederate server:

- List source trusted CAs and target key aliases
- Copy one or all trusted CAs from the source server
- Create certificates
- Create Certificate Signing Requests (CSRs)
- Import CA-signed and PKCS-12 certificates

### Using the Migration Tool

The configcopy tool may be used in conjunction with one or more property files to define the operational command and other parameters, including the source and/or target PingFederate servers, and to modify configuration settings as needed for the target environment.

Property-file templates are available for each command option in `<pf_install>/pingfederate/bin/configcopy_templates`.

**Note:** Refer to the README.txt file in the configcopy_templates directory for a list of all commands and summary information. See the template files themselves for parameters associated with each command (or with use cases), as well as lists of Override Properties (configuration settings that can be modified in transit), where applicable.

Copies of the templates can be configured as needed and then used together (or combined into one file). Use the applicable filenames as an argument when running...
configcopy.bat or configcopy.sh (depending on your operating system) for particular configurations, using the following command syntax:

(On Windows)

```bash
configcopy.bat
-Dconfigcopy.conf.file=<properties_file1>;<properties_file2>;
```

When paths are included with the filenames, you cannot use backslashes (`\`). Use forward slashes (`/`) or escape the backslash (`\\`).

(On Unix/Linux)

```bash
configcopy.sh
-Dconfigcopy.conf.file=<properties_file1>:
<properties_file2>:...
```

Note that the file separators are platform specific, corresponding to the syntax used for system-level path separators.

Alternatively (or in addition), you can specify any property values via command-execution arguments, using the following syntax:

```bash
configcopy[.sh] -D<property>=<value> ...
```

where `<property>` is any property named in the properties file and `<value>` is the value.

Command-line property designations take precedence over any values set in the properties file.

**Note:** Access to the Connection Management Web Services are password-protected (see “Authentication” on page 215). The usernames and passwords may be set in the properties file for both the source and target Web Services (passwords can be obfuscated). If passwords are set in the properties file, they cannot be overridden using the command line. If a password is not set, the `configcopy` tool prompts for it. Usernames always must be supplied where applicable, either in the command line or in the properties file.

The configcopy utility generates its own log file, `configcopy.log`, which is located in the `<pf_install>/pingfederate/log` directory. You can control settings for this log, as needed, in the file `configcopy.log4j.properties`, located in the `bin` directory.

**Caution:** Importing connections or other discrete configurations at the target server is not subject to the same rigorous data validation performed by the administrative console during manual configuration. Although some checks are made, it is possible to create invalid connections using the connection-migration process. Therefore, you should not use the configcopy tool to attempt to create settings at the target that do not exist at the source; for connections and other configurations copied separately, the tool is designed only for modifying the values of existing source settings to make them applicable to the target environment.

In addition, to avoid errors and prevent unstable target configurations due to missing components or faulty cross-component references (for example, invalid ID references from connection configurations to datastore configurations), be sure to adhere closely to the instructions provided in the following procedure.
Chapter 2
System Administration

To use configcopy:

1. Ensure access to the Connection Management Web Service is enabled for both the source and target PingFederate servers (see “Authentication” on page 215).

2. Determine which component configurations need to be copied, including plugins.

   For example, connection configurations always reference either adapter or token-translator configurations (or both) and may reference data-store configurations. These are all separate configurations, and must be copied separately (unless they already exist at the target) in conjunction with copying connection configurations.

   Server Settings, unless preconfigured at the target, also need to be copied over separately.

   Provisioning settings may be copied separately as needed to update target connections.

3. Determine whether any configuration property files or other supporting files need to be copied.

4. Ensure necessary plug-in JAR files are installed on the target server.

   The configcopy tool does not copy over these files, which include libraries for adapters, token translators, and JDBC or any custom database drivers.

   The JAR files are located in either:
   
   `<pf_install>/pingfederate/server/default/deploy`
   or:
   
   `<pf_install>/pingfederate/server/default/lib`

5. On the target server, ensure that signing certificates (or certificates used for XML decryption) are already in place (see “Security Management” on page 203).

   Private keys are not copied from server to server (public certificates may be copied); however, you may use configcopy to upload keys/certificates to the target server.

   Make note of identifying information about the target keys so you can reference the certificates in connection-copy properties.

6. If you have not yet installed your organization’s (CA-issued) SSL server certificate on both the target and source servers, either do so—you can use a configcopy command for this—or use one of the following work-arounds to ensure that configcopy can contact both servers:

   Either:
   
   - (Recommended) Install the Issuer certificate for the PingFederate SSL certificate in a separately managed trust store. Then the location of the file can be specified when running configcopy using the property `configcopy.connection.trust.keystore`.

   Or:
   
   - Install the Issuer certificate for the PingFederate SSL certificate into the trust store for the JDK under which configcopy runs.

   **Note:** If different SSL certificates are installed on the two servers, the configcopy tool must be able to trust both. In this case, both certificates must be installed in the trust store used by configcopy, or in the trust store for the JDK under which configcopy runs.
7. Create properties files for the necessary commands and associated command-parameter values needed to copy the required configurations and any additional files.

Refer to the README.txt file and to the properties-file templates in the directory: 
<pf_install>/pingfederate/bin/configcopy_templates

**Note:** This step and those following assume the use of properties files based on the templates provided; you may also use command-line parameters (see information earlier in this section).

8. If you are copying connections, override ID properties referencing adapter, data stores or other plug-in configurations, as needed (see Step 2).

**Important:** Ensure that the plug-in configurations are either previously defined at the target or are part of the same configcopy process used to copy the connections that depend on them.

9. Create a script or run a command (or command series) that executes configcopy for each of the prepared properties files.

See the discussion above for syntax requirements, or the README file.

---

### Outbound Provisioning CLI

PingFederate provides as command-line interface (CLI) to help manage automated outbound provisioning at IdP sites (see “Outbound Provisioning for IdPs” on page 34). Administrators can use this tool to view the status of user provisioning, either globally or one provisioning channel at a time, and to rectify unusual situations where provisioning at the service provider may get out of sync with the enterprise user store (see “Configuring Outbound Provisioning” on page 315).

The CLI tool, provmgr.bat or provmgr.sh, is located in the directory <pf_install>/pingfederate/bin. The tool interacts with the internal data store PingFederate uses to maintain provisioning synchronization between the LDAP user store and the target service (see “Configuring Outbound Provisioning Settings” on page 114).

Note that the tool creates its own log file, provmgr.log, located in the directory <pf_install>/pingfederate/log. You can control settings for this log, as needed, in the file provmgr.log4j.properties, located in the bin directory.

The following tables describes the available global and channel-specific command arguments:

**Table 14:** Outbound Provisioning CLI Global Options

<table>
<thead>
<tr>
<th>Command Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--help</td>
<td>Describes the available options. The help is also displayed if the command is run with no arguments.</td>
</tr>
</tbody>
</table>
Table 14: Outbound Provisioning CLI Global Options (Continued)

<table>
<thead>
<tr>
<th>Command Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--show-channels</td>
<td>Lists all channels in a table format, showing for each:</td>
</tr>
<tr>
<td></td>
<td>• ID - A numeric channel ID (channel-specific commands need this ID)</td>
</tr>
<tr>
<td></td>
<td>• Name - The channel name</td>
</tr>
<tr>
<td></td>
<td>• Connection ID</td>
</tr>
<tr>
<td></td>
<td>• Status (active/inactive) - Both the connection and the channel status are shown (see “Channel Activation and Summary” on page 327)</td>
</tr>
<tr>
<td></td>
<td>• User count/dirty-user-record count (e.g.: 5000/12 means 5000 users and 12 dirty records)</td>
</tr>
<tr>
<td></td>
<td>• Source (as LDAP URL)</td>
</tr>
<tr>
<td></td>
<td>• Target code</td>
</tr>
<tr>
<td>--show-nodes</td>
<td>Shows all the provisioning-server nodes with their status and the last timestamp (applies only to a failover configuration—see the PingFederate Server Clustering Guide).</td>
</tr>
<tr>
<td>--force-node-backup</td>
<td>Use with node number:</td>
</tr>
<tr>
<td></td>
<td>-n &lt;node ID&gt;</td>
</tr>
<tr>
<td></td>
<td>Sets the provisioner mode to FAILOVER for the associated PingFederate server node (see the Server Clustering Guide).</td>
</tr>
</tbody>
</table>

The following table describes the available channel-specific command arguments:

Note: With each command, specify the channel with the argument:
- `-c <channel-id-number>`

Example:
```
provmgr -c 1 --show-source
```

You can determine channel ID numbers by using the global command:
```
provmgr --show-channels
```
Table 15: Outbound Provisioning CLI Channel-Specific Options

<table>
<thead>
<tr>
<th>Command Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--reset-group-timestamp</td>
<td>Deletes the user-group timestamp, which forces the provisioner to process the provisioning group on the next cycle, even if the timestamp on that group did not actually change.</td>
</tr>
<tr>
<td></td>
<td>Depending on your LDAP server and administrative practices, you may want to schedule this command to run periodically to catch up with any users that may have been deleted (rather than deactivated) in the directory server: some directory servers do not update the group timestamp for deleted users.</td>
</tr>
<tr>
<td></td>
<td><strong>Important</strong>: This option should seldom be needed if users are deactivated rather than deleted. If it is needed, you may wish to schedule it when other network activity is low.</td>
</tr>
<tr>
<td>--reset-attribute-sync</td>
<td>Sets the attribute sync timestamp to 1, which forces the provisioner to look at all users for changes, not only those that have a newer timestamp on their LDAP entry.</td>
</tr>
<tr>
<td></td>
<td><strong>Important</strong>: This option should be needed rarely and may consume considerable network resources, depending on the number of users. If it is needed, you may wish to schedule it when other network activity is low.</td>
</tr>
<tr>
<td>--reset-values-hash</td>
<td>Removes the values hash for all users. (The database stores a hash of attribute values for users to determine whether any values have been changed.)</td>
</tr>
<tr>
<td></td>
<td>This argument forces users that have a newer timestamp on their LDAP entry to be updated at the service provider, regardless of the actual field values. Note, however, that users whose recorded timestamp is unchanged are not updated.</td>
</tr>
<tr>
<td>--reset-all</td>
<td>Equivalent to using all three of the arguments above.</td>
</tr>
<tr>
<td></td>
<td><strong>Important</strong>: This option should be needed rarely if ever and may consume considerable network resources, depending on the number of users. If it is needed, you may wish to schedule it when other network activity is low.</td>
</tr>
<tr>
<td>--show-dirty-records</td>
<td>Lists all users or groups that have not been provisioned or updated at the service-provider site.</td>
</tr>
<tr>
<td>--show-dirty-group-records</td>
<td>List groups that have not been provisioned or updated at the service-provider site.</td>
</tr>
<tr>
<td>--show-dirty-user-records</td>
<td>List all users that have not been provisioned or updated at the service-provider site.</td>
</tr>
</tbody>
</table>
### Table 15: Outbound Provisioning CLI Channel-Specific Options (Continued)

<table>
<thead>
<tr>
<th>Command Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--show-group</td>
<td>Shows all internal database fields related to the specified user or group, including transitory mapping fields (fields waiting to be pushed to the service provider); for a user, shows all LDAP attributes retrieved from the directory server.</td>
</tr>
<tr>
<td>--show-user</td>
<td>Note: You can obtain user or group names and GUIDs for dirty records, as needed, using any of the --show-dirty-* options (described above). The LDAP GUID, if used and if it is binary, should be entered in hexadecimal format (as shown in log files).</td>
</tr>
<tr>
<td>Use with:</td>
<td>Examples:</td>
</tr>
<tr>
<td>-u &lt;provider name&gt;</td>
<td>provmgr.sh --show-user -u john@example,com</td>
</tr>
<tr>
<td>Or:</td>
<td>provmgr.sh --show-user -g ffd448643f812b43a0bee2504173f0</td>
</tr>
<tr>
<td>-g &lt;LDAP GUID&gt;</td>
<td>--clear-dirty-records Clear the dirty flag on all records.</td>
</tr>
<tr>
<td></td>
<td>--clear-dirty-group-records Clears the dirty flag on all group records.</td>
</tr>
<tr>
<td></td>
<td>--clear-dirty-user-records Clears the dirty flag on all user records.</td>
</tr>
<tr>
<td></td>
<td>--delete-dirty-records Removes all dirty records from the internal store.</td>
</tr>
<tr>
<td></td>
<td>--delete-dirty-group-records Removes all dirty group records from the internal store.</td>
</tr>
<tr>
<td></td>
<td>--delete-dirty-user-records Removes all dirty user records from the internal store.</td>
</tr>
<tr>
<td></td>
<td>--delete-all The delete-all parameter removes all users and groups from the internal store and deletes the provisioning group timestamp and the last attribute-sync timestamp. The delete-all-users parameter deletes users and timestamps but retains groups.</td>
</tr>
<tr>
<td></td>
<td>The effect of either command is to reset the channel to its initial state for user provisioning. All user metadata is lost and provisioning for the channel will start from the beginning, picking up all users (and groups if deleted) and pushing them to the service provider when the synchronization frequency interval is expired (see “Configuring Outbound Provisioning Settings” on page 114).</td>
</tr>
<tr>
<td></td>
<td>Important: These options should be needed rarely if ever. If needed, you may wish to schedule the operation when other network activity is low.</td>
</tr>
<tr>
<td></td>
<td>--delete-all-users</td>
</tr>
<tr>
<td></td>
<td>--show-target Displays the target configuration.</td>
</tr>
<tr>
<td></td>
<td>--show-source Displays all source LDAP configuration parameters, including settings and location.</td>
</tr>
</tbody>
</table>
Customizing User-Facing Screens

PingFederate supplies HTML templates to provide information to the end user or to request user input during SSO/SLO processing. These template pages utilize the Velocity template engine, an open-source Apache project, and are located in the `<pf_install>/pingfederate/server/default/conf/template` directory.

You can modify most of these pages in a text editor to suit the particular branding and informational needs of your PingFederate installation. (Cascading style sheets and images for these pages are included in the `template/assets` subdirectory.) Each page contains both Velocity constructs and standard HTML. The Velocity engine interprets the commands embedded in the template page before the HTML is rendered in the user's browser. At runtime, the PingFederate server supplies values for the Velocity variables used in the template.

Each sample template provided contains specific variables that can be used for rendering the associated Web-browser page. You can see the variables and usage examples in the comments of each template. The following table describes variables that are available across all templates.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$escape</td>
<td>A utility class that can be used to escape String variables inserted into the template, for example, <code>$escape.escape($client.name)</code></td>
</tr>
<tr>
<td>$HttpServletRequest</td>
<td>A Java object instance of <code>javax.servlet.http.HttpServletRequest</code>. Used to add additional knowledge about the request that is otherwise unavailable in the template (for example, HTTP User-Agent header).</td>
</tr>
<tr>
<td>$HttpServletResponse</td>
<td>A Java object instance of <code>javax.servlet.http.HttpServletResponse</code>. Used to modify the response in the template (for example, setting additional browser cookies).</td>
</tr>
<tr>
<td>$locale</td>
<td>A Java object instance of <code>java.util.Locale</code> that represents a user’s country and language. Used to customize the end-user experience. For example, the locale is used to display content in the user’s preferred language.</td>
</tr>
<tr>
<td>$PingFedBaseURL</td>
<td>The PingFederate base URL (see “Specifying Federation Information” on page 109”).</td>
</tr>
<tr>
<td>$templateMessages</td>
<td>Used to localize messages in the template (based on user’s Locale), an instance of <code>com.pingidentity.sdk.locale.LanguagePackMessages</code>. For more information, see the Javadoc for the <code>LanguagePackMessages</code> class in the directory <code>&lt;pf_install&gt;/pingfederate/sdk/doc</code>.</td>
</tr>
<tr>
<td>$TrackingId</td>
<td>The user’s session tracking ID (see the last “Note” on page 45 in the section “Managing Log Files”).</td>
</tr>
</tbody>
</table>

For information about Velocity, please refer to the Velocity project documentation on the Apache Web site.
Changing Velocity or Javascript code is not recommended.

At runtime, the user's browser is directed to the appropriate page, depending on the operation being performed and where the related condition occurs (see tables below). For example, if an SSO error occurs during IdP-initiated SSO, the user's browser is directed to the IdP's SSO error-handling page.

Applications can override the PingFederate server-hosted pages provided specifically for SSO and SLO errors by specifying a URL value in the relevant endpoint’s InErrorResource parameter (see “Application Endpoints” on page 529). Administrators can override SSO/SLO success pages by specifying default URLs in the administrative console (for the IdP configuration, see “Configuring a Default URL and Error Message” on page 243; for the SP, see “Configuring Default URLs” on page 346).

The following tables describe each of the templates. To help identify the templates from the end user's point of view, the tables are organized by the titles that appear at the top of the user's browser window.

**Note:** The titles listed in the tables are the defaults shipped with PingFederate and may be modified. The templates retrieve titles and other text from a language localization file (see “Localization” on page 92).

### IdP User-Facing Pages

#### Table 16: IdP User-Facing Pages

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Change Password</strong>&lt;br&gt;html.form.change.password.template.html</td>
<td>Allows a user to change his or her password via the HTML Form Adapter (see “Configuring the HTML Form IdP Adapter” on page 515).</td>
<td>Normal</td>
<td>User input required</td>
</tr>
<tr>
<td><strong>Change Password Message</strong>&lt;br&gt;html.form.message.template.html</td>
<td>Displayed when a user has successfully changed his or her password (see “Configuring the HTML Form IdP Adapter” on page 515).</td>
<td>Normal</td>
<td>User input required</td>
</tr>
<tr>
<td><strong>Error - Single Logout</strong>&lt;br&gt;idp.slo.error.page.template.html</td>
<td>Displayed when an SLO request fails and no other SLO error landing page is specified.</td>
<td>Error</td>
<td>User should close browser</td>
</tr>
<tr>
<td><strong>Error - Single Sign-On</strong>&lt;br&gt;idp.sso.error.page.template.html</td>
<td>Displayed when IdP-initiated SSO fails on the IdP side and no other SSO error landing page is specified. Displays system errors and information for the user.</td>
<td>Error</td>
<td>Consult log and Web developer</td>
</tr>
<tr>
<td><strong>IdP Logout</strong>&lt;br&gt;idp.logout.success.page.template.html</td>
<td>Default page may be displayed when user is logged out of the IdP by the HTTP Basic and HTML Form adapters (see “Configuring the HTTP Basic IdP Adapter” on page 509 and “Configuring the HTML Form IdP Adapter” on page 515).</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>
### Table 16: IdP User-Facing Pages (Continued)

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
</table>
| **IdP Logout Confirmation**  
idp.slo.confirm.page.template.html | Displayed to prompt the user to confirm Single Logout (SLO). For more information, see “Configuring a Default URL and Error Message” on page 243. | Normal | User input required |
| **Password Management System Message**  
html.form.message.template.html | Displayed when a user is being redirected to a password management system to change his or her password (see “Configuring the HTML Form IdP Adapter” on page 515). | Normal | User input required |
| **Please Select Authentication System**  
sourceid-choose-idp-adapter-form-template.html | Displayed when the user must choose from several IdP security domains. Based on the user’s selection, the server redirects the browser to the appropriate adapter instance for authentication. | Normal | User must make selection |
| **Login Challenge**  
html.form.login.challenge.template.html | Displays a configurable challenge form for two-step authentication. This template can be used, for example, to create a RADIUS challenge form when using the RADIUS Username/Password Credential Validator (see “Configuring the HTML Form IdP Adapter” on page 515). | Normal | User input required |
| **Sign On**  
html.form.login.template.html | Displays a configurable user logon form when the HTML Form Adapter is in use for a connection (see “Configuring the HTML Form IdP Adapter” on page 515). | Normal | User input required |
| **Signed Out**  
sourceid-wsfed-idp-signout-cleanup-template.html | Indicates user signed out of the IdP under the WS-Federation protocol and lists each successful SP logout, when applicable. Also displays when an OpenID Connect Client sends a logout request to the /idp/startSLO.ping endpoint (see “Asynchronous Front-Channel Logout” on page 157). | Normal | None |
| **Signing Out**  
sourceid-wsfed-idp-signout-cleanup-invisible-template.html | WS-Federation and OpenID Connect Client IdP sign-out processing page. **Note:** No HTML is rendered in the browser. | Normal | None |
| **Success - Single Logout**  
idp.slo.success.page.template.html | Displayed when an SLO request succeeds but no other SLO landing page is specified. | Normal | None |
| **User Consent**  
consent-form-template.html | Displayed when a request requires a user’s consent for an SSO to an SP. | Normal | User input required |
Table 16: IdP User-Facing Pages (Continued)

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working . . . sourceid-wsfed-http-post-template.html</td>
<td>Used to auto-submit a WS-Federation assertion to the SP. If Javascript is disabled, the user is prompted to click a button to POST the assertion directly. <strong>Note:</strong> Normally not displayed if Javascript executes properly.</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>

**SP User-Facing Pages**

Table 17: SP User-Facing Pages

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Link Removed TerminateAccountLinks.page.template.html</td>
<td>Communicates a user’s successful “defederation” operation.</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Account Linking LocalIdPasswordLookup.form.template.html</td>
<td>Used to authenticate a user at the SP when an account link needs to be established.</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Authentication Failed sourceid-wsfed-idp-exception-template.html</td>
<td>Displayed when an authentication challenge fails during WS-Federation processing.</td>
<td>Error</td>
<td>Consult log and Web developer</td>
</tr>
<tr>
<td>Error - Single Logout sp.slo.error.page.template.html</td>
<td>Displayed when an SLO request fails and no other SLO error landing page is specified.</td>
<td>Error</td>
<td>User should close the browser</td>
</tr>
<tr>
<td>Error - Single Sign-On sp.sso.error.page.template.html</td>
<td>Displayed when SP-initiated SSO fails (or IdP-initiated SSO fails on the SP side) and no other SSO error landing page is specified.</td>
<td>Error</td>
<td>Consult log and Web developer</td>
</tr>
<tr>
<td>Select Identity Provider sourceid-saml2-idp-selection-template.html</td>
<td>The user requested SP-initiated SSO, but the IdP partner was not specified in the appropriate query parameter or cookie. This page allows the user to select the IdP manually. Based on the user’s selection, the server redirects the browser to the appropriate IdP partner’s SSO service.</td>
<td>Normal</td>
<td>User must make selection</td>
</tr>
<tr>
<td>Signed Out - Service Provider sourceid-wsfed-sp-signout-cleanup-template.html</td>
<td>Displays the user’s sign-out status.</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Success - Single Logout sp.slo.success.page.template.html</td>
<td>Displayed when an SLO request succeeds and no other SLO success landing page is specified.</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>
### Table 17: SP User-Facing Pages (Continued)

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Sign-On Target Unspecified &lt;br&gt;sp.sso.success.page.template.html</td>
<td>Displayed when an SSO request succeeds but no target-resource parameter is specified by the incoming URL, and no default URL is set (see “Configuring Default URLs” on page 346).</td>
<td>Error</td>
<td>Consult Web developer, or specify default URL</td>
</tr>
</tbody>
</table>

### Either IdP or SP User-Facing Pages

### Table 18: Either IdP or SP User-Facing Pages

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error - Single Sign-On &lt;br&gt;generic.error.msg.page.template.html</td>
<td>For an Auto-Connect SSO transaction, indicates a range of possible error conditions (see “Using Auto-Connect” on page 30):&lt;br&gt;• The requesting Auto-Connect partner is not found in the PingFederate server’s list of allowed domains.&lt;br&gt;• The partner’s metadata is not accessible.&lt;br&gt;• The server is not configured for Auto-Connect.&lt;br&gt;• General error, with error code.</td>
<td>Error</td>
<td>Consult log, check configuration, or contact partner. If unresolved, contact Ping Identity support.</td>
</tr>
<tr>
<td>Error &lt;br&gt;generic.error.page.template.html</td>
<td>Indicates that an unknown error has occurred and provides an error number and (optionally) an error message.</td>
<td>Error</td>
<td>Consult log, contact Ping Identity support</td>
</tr>
<tr>
<td>Multiple SSO in Progress &lt;br&gt;speed.bump.template.html</td>
<td>Displayed to a user when response is slow due to simultaneous SSO requests coming from multiple browser tabs.</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Page Expired &lt;br&gt;state.not.found.error.page.template.html</td>
<td>Displayed when simultaneous SSO requests from multiple tabs using the same key value cause a user session to be overwritten or deleted and remaining requests attempt to retrieve the state fail.</td>
<td>Error</td>
<td>None</td>
</tr>
<tr>
<td>Sign On &lt;br&gt;AbstractPasswordIdpAuthnAdapter.form.template.html</td>
<td>Challenges user for credentials when authentication can take place via HTTP Basic Authentication or an HTML form, depending on the operational mode.</td>
<td>Normal</td>
<td>User must sign on</td>
</tr>
</tbody>
</table>
Chapter 2
System Administration

Table 18: Either IdP or SP User-Facing Pages (Continued)

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submit Form</td>
<td>Whenever the server posts a form, this template is used to auto-submit the form. If Javascript is disabled, the user is prompted to click a button to post the form manually. <strong>Note</strong>: Normally not displayed if Javascript executes properly.</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>

OAuth User-Facing Pages

Table 19: OAuth User-Facing Pages

<table>
<thead>
<tr>
<th>Page Title and template file name</th>
<th>Purpose</th>
<th>Type</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Revocation</td>
<td>Provides a means for end-users (resource owners) to revoke persistent access grants.</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Information Access Approval</td>
<td>Advises resource owners that their information is being requested by the identified OAuth client and provides for approval/disapproval. This page appears for Implicit or Authorization Code grant types, either one time only or repeatedly depending on OAuth AS settings (see “Authorization Server Settings” on page 159). <strong>Tip</strong>: The OAuth client configuration provides an option to bypass this approval page entirely, as needed for trusted clients (see &quot;Configuring a Client&quot; on page 178).</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>

Localization

The English display text in each of the user-facing templates is retrieved from a default localization file, `pingfederate-messages.properties`, in the directory:

```
<pf_install>/pingfederate/server/default/conf/language-packs
```

An administrator can localize the text by copying the default file, providing translated text in place of English, and then adding the standard language tag to the base filename (as indicated in browser settings).

For example, for text in French rename the translated copy of the localization file in the `language-packs` directory to:

```
pingfederate-messages_fr.properties
```
To include a region, append the capitalized abbreviation using an underscore.

**Note:** The capitalization and underscore usage may not correspond to the way regions are listed in browser settings. However, the usage is required by the Java-based localization implementation (see “Retrieving Localized Messages” on page 93).

For example, for Canadian French:

```
pingfederate-messages_fr_CA.properties
```

**Note:** If the system language of the PingFederate server is not English, copy `pingfederate-messages.properties` as `pingfederate-messages_en.properties` for the English users (if any), translate `pingfederate-messages.properties` for the local users, and provide additional translations as needed.

Developers can also customize the look and feel of the templates by using localization variables in logic statements to control fonts, color, and other style elements. Refer to the templates for examples.

**Tip:** To maximize performance, PingFederate caches localized UI strings on start-up. For testing new localization implementations, an administrator can temporarily turn off caching by changing the value of `cache-language-pack-messages` to `false` in:

```
.../pingfederate/server/default/data/config-store/locale-options.xml
```

Be sure to return the value to `true` when testing is complete. Note that restarting the server is required for changes to configuration files.

### Overriding Locales with Cookies

For convenience, an administrator or Web developer might want to provide end-users a means of overriding browser language preferences temporarily by setting cookies—for example, by creating a company Web portal link for users to click instead of manually changing their browser options.

By default, PingFederate supports overriding the locale via a cookie named `pf-accept-language`. The cookie value must conform to guidelines defined under IETF BCP 47. For more information, see documentation for the Java core method `Locale.forLanguageTag(String languageTag)`. (This locale-override behavior is the default implementation of the Java interface `LocaleOverrideService` defined in the PingFederate SDK. For more information, see the Javadoc for that interface in the directory `<pf_install>/pingfederate/sdk/doc`.)

PingFederate displays the language indicated in the cookie if the language is supported in the language-packs directory. If the matching localization file is not found, PingFederate defaults to the browser settings.

### Retrieving Localized Messages

Retrieval of localized messages is supported via the `LanguagePackMessages` class available in the PingFederate SDK. An instance of this class is passed into every template file and available for use there. For information, refer to the Javadoc for the class in the directory `<pf_install>/pingfederate/sdk/doc`.  

Note: The capitalization and underscore usage may not correspond to the way regions are listed in browser settings. However, the usage is required by the Java-based localization implementation (see “Retrieving Localized Messages” on page 93).
Chapter 2
System Administration

Configuring Password Policy

PingFederate applies a configurable policy to passwords, pass phrases, and shared secrets defined by the administrators in the administrative console. These fields include, but are not limited to:

- Passwords used by HTTP Basic authentication for:
  - Inbound SOAP messages from partners via back-channel calls
  - WS-Trust STS
- Shared secrets used by the credentials defined for:
  - Attribute Query
  - JMX
  - Connection Management
  - SSO Directory Service
- Passwords used by instances of the Simple Username Password Credential Validator
- Passwords used for encrypting certificates exported with their private keys
- Pass phrases used by IdP Discovery
- Passwords used by administrative console credentials when native authentication is used

To configure the complexity of the password policy:

2. Save the changes.
3. Restart PingFederate.

For a clustered PingFederate environment, perform these steps on the console node. No changes or restart of PingFederate is required on the engine nodes.

Note: Passwords external to PingFederate—passwords used by instances of the Data Stores, for example—are not subject to this password policy.

Extending the Lifetime of the PF Cookie

PingFederate identifies sessions by their respective PF cookie. By default, the PF cookie is a session cookie. You can extend the lifetime of the PF cookie by making it a persistent cookie. Unlike session cookies, persistent cookies are saved to disk, enabling your web browser to reuse them when restarted.

To extend the lifetime of the PF cookie in a standalone environment:

2. Modify the `cookie-max-age` value. The default value (-1) makes the cookie a session cookie; a positive integer defines the age of the persistent cookie in seconds.
3. Save the change.
4. Restart PingFederate.
Adding Custom HTTP Response Headers

To extend the lifetime of the PF cookie in a clustered PingFederate environment:

2. Modify the `cookie-max-age` value. The default value (-1) makes the cookie a session cookie; a positive integer defines the age of the persistent cookie in seconds.
3. Save the change.
4. Log on to the administrative console.
5. Click **Cluster Management** on the Main Menu.
6. On the Cluster Management screen, click **Replicate Configuration**.

**Note:** It is not necessary to restart PingFederate on any running engine node.

The HTML Form IdP Adapter utilizes the PF cookie to manage its sessions. You can bind the lifetime of the authenticated sessions to the age of the persistent PF cookie. For more information, see "Configuring the HTML Form IdP Adapter" on page 515.

Adding Custom HTTP Response Headers

The PingFederate administrative console and runtime server are capable of returning custom HTTP response headers, such as Strict-Transport-Security to enforce HTTPS based access and P3P for Microsoft Internet Explorer interoperability.

To add custom HTTP response headers:

2. Save the changes.
3. Restart PingFederate.

In a clustered PingFederate environment:

1. On the console node, edit `response-header-admin-config.xml` or `response-header-runtime-config.xml` on the console node.
2. Restart PingFederate.
3. Log on to the administrative console.
4. Click **Cluster Management** on the Main Menu.
5. On the Cluster Management screen, click **Replicate Configuration**.
6. For each engine node, restart PingFederate as `response-header-admin-config.xml` and `response-header-runtime-config.xml` are not reloaded after Replicate Configuration.
The System Settings links on the Main Menu (under Server Configuration) provide access to global settings that may apply to either an IdP or an SP configuration.

This chapter covers:

- “Managing Server Settings” on page 97
- “Managing Data Stores” on page 117
- “Configuring IdP Discovery” on page 133
- “IdP-to-SP Adapter Mapping” on page 137
- “STS Token Exchange Mapping” on page 146

Note: The information in this chapter is presented from the viewpoint of an administrative user with “Admin” permissions (see “Account Management” on page 65).

Managing Server Settings

Server settings include unique federation server identifiers, the designation of your site’s federation role (SP, IdP, or both), and your enabled federation protocols (see the “Supported Standards” chapter in Getting Started).

Server settings also include system-administration configuration (one-user or multi-user), email notification options and setup, and a shortcut link to account management (when multi-user administration is enabled).

If you have enabled Auto-Connect and/or Outbound Provisioning, then you must configure several parameters specific to those features in the System Settings task flow.
You configure many of these settings initially during the installation setup (see "Running PingFederate for the First Time" in the “Installation” chapter of Getting Started), but you can change or add to them as needed from the Main Menu.

**Note:** For information about WS-Trust STS Settings, see “Configuring STS Authentication” on page 438.

Information in this section covers:
- “Setting Administration Options” on page 98
- “Entering System Information” on page 99
- “Configuring Runtime Notifications” on page 99
- “Configuring Runtime Reporting” on page 100
- “Managing Accounts” on page 105
- “Choosing Roles and Protocols” on page 106
- “Specifying Federation Information” on page 109
- “Setting System Options” on page 111
- “Configuring Outbound Provisioning Settings” on page 114
- “Configuring Auto-Connect Metadata Signing” on page 115
- “Configuring Auto-Connect Metadata Lifetime” on page 116
- “Saving and Editing Server Settings” on page 117

### Setting Administration Options

On the System Administration screen, PingFederate provides a choice of single- or multi-user access to the administrative console.

**Note:** If you are using your network’s LDAP user-data store or client certificates for administrative-console authentication, this screen is not presented (see “Alternative Console Authentication” on page 68).

If you choose Single-user Administration, the console is accessible only by using the default Administrator ID, for which full privileges are provided. Multi-user Administration (the default) provides role-based access control (see “Account Management” on page 65).

**Tip:** To return to single-user administration after having previously enabled multi-user, only one user can be marked as active under Account Management.
Managing Server Settings

**Entering System Information**

On the System Info screen, you provide general information about your company.

![System Info Screen](image)

To reach this screen:
1. Click **Server Settings** on the Main Menu.
2. Click **System Info** under the Server Settings tab.

**Configuring Runtime Notifications**

Depending on your licensing agreement, your PingFederate license may have an expiration date. Under Runtime Notifications you can set up the server to send an email warning when your license is about to expire.

*Note:* The license-notification option does not appear if you have a perpetual license.

You can also configure the server to send an email notification to a specific administrator (or a group) when a certificate used by PingFederate is about to expire, or has expired.

*Note:* When a certificate expires, PingFederate always writes an error in the server log, regardless of whether runtime notification is configured (see “Managing Log Files” on page 44).
Chapter 3  
System Settings

To reach this screen:
1. Click **Server Settings** on the Main Menu.
2. Click **Runtime Notifications** on the Summary screen.

To configure notifications:
1. Select the checkbox next to the type of notification you want, and enter an email address.
2. If you are configuring certificate-expiration notification, enter an advance-warning time period in the Initial Warning field (optional) and in the Final Warning field.

   **Note:** When advance certificate-expiration notification is configured, the server also sends notification if a license expires.

3. If you have not previously configured PingFederate to access your email server, click **Email Server Settings** (see “Managing Email Configuration” on page 71).

Configuring Runtime Reporting

PingFederate supports runtime monitoring and reporting through the Simple Network Management Protocol (SNMP), a standard used by network-management consoles to monitor network and server activity across an enterprise.

PingFederate also supports runtime monitoring and reporting through Java Management Extensions (JMX) (see “Runtime Monitoring Using JMX” on page 102).

Using SNMP Monitoring

The SNMP Management Information Base (MIB) defines network data available for SNMP monitoring. The MIB file is located in:

```<pf_install>/pingfederate/SNMP```

The MIB describes the object identifiers that PingFederate uses to communicate information through SNMP. These identifiers are globally unique and managed by the Internet Assigned Numbers Authority (IANA).
Configure access to SNMP monitoring on the Runtime Reporting screen.

SNMP supports Gets and Traps. A Get is a request for status information sent by a network-management console to an SNMP agent. Embedded within each PingFederate server is an SNMP agent that brokers the communication between the management console and the PingFederate runtime engine (for each engine separately when PingFederate is deployed in a cluster—see the PingFederate Server Clustering Guide).

**Gets**  
PingFederate responds to two SSO/SLO types of Get requests:
- The total number of transactions that the server instance has processed since installation
- The total number of failed transactions that the server instance has encountered since installation

In addition, because PingFederate is built within an existing Jetty framework, Gets include a variety of server information available via Jetty-standard Managed Beans (MBeans). A detailed list of this information is provided in the MIB file in the pingfederate/SNMP directory. (For more information about MBeans, see the next section, “Runtime Monitoring Using JMX”).

**Note:** Some operating systems (Solaris 10, for example) may not allow the SNMP agent to bind to privileged ports: those below 1024. Consult your operating system’s documentation on how to get around this limitation, or change the default port 161 to a port above 1023.

**Traps**  
A Trap is a spontaneous communication from an agent to a network-management console. PingFederate generates a Trap at regular intervals—the server “heartbeat.” Each Trap contains the amount of time the server instance has been running since its most recent start-up.

- If you configure Traps, change settings as needed and then click Test SNMP Configuration to send a single Trap to your network-management console.

You can also use an HTTP call at any time to verify that the PingFederate server is running (see “System-Services Endpoints” on page 540).

**Runtime Monitoring Using JMX**

Similar to SNMP, JMX technology represents a Java-centric approach to application management and monitoring. JMX exposes instrumented code in the form of
MBeans. Application management systems that support JMX technology—for example, the standard JDK client JConsole—may request runtime information from the PingFederate JMX server.

PingFederate’s JMX server reports monitoring data for SSO and SLO transactions. In addition, as with SNMP monitoring, numerous Jetty-standard MBeans are available to the PingFederate server’s JMX clients (see “Example Jetty Metrics” on page 104).

**Important:** Authentication is required for JMX-client access to PingFederate runtime data (see “Authentication” on page 215).

**Tip:** Administrators can supplement JMX monitoring information by applying third-party analysis and reporting tools to the security audit log (see “Security Audit Logging” on page 48). That log records fine-grain details (including, for example, response times and event types) for all server transactions.

### SSO-SLO Monitoring

For SSO/SLO transaction processing, PingFederate provides these MBeans:

- pingfederate:type=TOTAL_FAILED_TRANSACTIONS
- pingfederate:type=TOTAL_TRANSACTIONS

Each type contains a single attribute, Count, which reports the same information as an SNMP Get (see the previous section, “Configuring Runtime Reporting” on page 100).

### Provisioning Monitoring

**Important:** Monitoring Outbound Provisioning transactions using JMX has been deprecated. By default, PingFederate logs outbound provisioning events to provisioner-audit.log (see “Outbound Provisioning Audit Logging” on page 50).

**Note:** Repeat these steps on each PingFederate server configured to process Outbound Provisioning (see “Outbound Provisioning for IdPs” on page 34).

When JMX monitoring for Outbound Provisioning is enabled, PingFederate provides an MBean called pf.provisioning:type=saas.provisioning.events. The MBean exposes five JMX Operations, each corresponding to the Java methods described in the following table. Each method returns a CompositeData object,
which allows for the retrieval of complex data without requiring application-specific code to reside with the JMX client.

### Table 20 Outbound Provisioning JMX Monitoring Options

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Parameter</th>
</tr>
</thead>
</table>
| `viewEvents`                  | Gets an array of specific events based on the given criteria. The parameters filter the data collectively; that is, they are joined logically by “and”. | `wasSuccessful` – If true, returns information only on successful transactions; false returns information only on failed transactions; null returns all transactions.  
`eventTypeStr` – The type of event. Valid values are: CREATE, UPDATE, DISABLE, ENABLE; null or an empty string returns all types.  
`fromDate` – See Note below.  
`toDate` – See Note below. |
| `eventSummaryReport`          | Gets a summary of transactions counts for the given time period. Counts are provided for success, failure, and total. Each count includes a drill-down capability, providing counts by event type. | See Note below. |
| `provisioningCycleSummary`    | Gets a total count of provisioning cycles for the given time period. The drill-down provides information for each cycle, including success totals for users and groups added, modified, or removed in the internal tracking database; for the target data store, success and failure totals are listed for each type of transaction. | See Note below. |
| `eventSummaryReportAllData()` | Gets a summary of transaction counts with no time constraints (equivalent to `eventSummaryReport` with null or empty strings used as parameters). | None. |
| `eventSummaryRollup()`        | Gets a report representing an aggregate of multiple Summary Reports covering the last 0, 1, 2, 7, 30, 60, 90, 180, and 360 days. | None. |

**Note:** Date parameters may be formatted as either `yyyy`, `yyyy-MM-dd`, or `yyyy-MM-dd HH:mm:ss`. A null value or empty string for a date parameter indicates no constraint for that end of the range.
Example Jetty Metrics

The following table describes examples of Jetty MBean metrics, available via JMX, that administrators may find useful to supplement information provided via the PingFederate-specific MBeans described in previous sections.

<table>
<thead>
<tr>
<th>MBean</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.pingidentity.appserver.jetty.server.connector.ssl.nio:runtimesslselectchannelconnector</td>
<td>connections – The total number of TCP connections accepted by the server. connectionsDuration* – How long connections are kept open. Maximum, mean, standard deviation, and total accumulated time are available. connectionsOpen – The current number of open connections. Maximum is also available [connectionsOpenMax]. connectionsRequests* – Number of requests processed per connection. Maximum, mean, and standard deviation are available.</td>
</tr>
<tr>
<td>org.eclipse.jetty.server.handler:statisticshandler</td>
<td>requests – Total number of requests received. requestsActive – Number of requests currently being processed. Max is also available. requestTime – Request duration. Maximum, mean, standard deviation, and total accumulated time are available. responses1xx, responses2xx, responses3xx, … – Total number of requests that returned HTTP status codes of 1xx, 2xx, 3xx, etc.</td>
</tr>
<tr>
<td>org.eclipse.jetty.util.thread:queuedthreadpool</td>
<td>idleThreads – Number of idle threads currently available. threads – Number of threads currently running (including both idle and active). minThreads – Minimum number of threads in the pool. maxThreads – Maximum number of threads in the pool. lowOnThreads – A boolean flag indicating whether the pool is running low on threads.</td>
</tr>
</tbody>
</table>
Advanced JMX Configuration

By default, PingFederate uses port 1099 for its JMX server. To change the port or other JMS configuration items, if needed, modify the configuration file `jmx-remote-config.xml` in the directory `<pf_install>/server/default/conf`.

**Note:** When connecting to the JMX service using SSL (the default), ensure that the client trusts the PingFederate SSL server certificate presented (see “SSL Server Certificates” on page 204). (This should be a consideration only during testing, when using the certificate installed with PingFederate or another self-signed certificate.)

Managing Accounts

When you choose multi-user system administration, you can create users during installation or while configuring Server Settings (see “Setting Administration Options” on page 98).

Alternatively, you can set up and maintain user accounts later as a separate task (assuming you have user administration permissions—see “Account Management” on page 65). By default for installation, the user “Administrator” has full system permissions.

- To continue, click **Next** or **Save**.
- For information about adding or managing users, see “Account Management” on page 65.

**Note:** If you are using your network’s LDAP user-data store or client certificates for PingFederate authentication, the Account Management screen is not presented (see “Alternative Console Authentication” on page 68).
Choosing Roles and Protocols

On the Roles and Protocols screen, select which role(s) your organization plays and which sets of standards you will use with your PingFederate server (see the “Supported Standards” chapter in Getting Started).

**Note:** If you are using the PingFederate WS-Trust STS for either an IdP, an SP, or both, notice that a new configuration step, WS-Trust STS Settings, appears under the Server Settings tab. For information about this configuration, see “WS-Trust STS Configuration” on page 437.

Also on this screen, you can choose any of several options:

- Enable the PingFederate OAuth AS (see “About OAuth” on page 10).
  
  You can also extend OAuth processing to OpenID Connect policy configurations (see “OpenID Connect” on page 14).

- As an SP, if you are using SAML 2.0 XASP for multiple IdP connections, you may choose to have PingFederate determine dynamically which connection to use (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

- For either role you can enable Auto-Connect for SAML 2.0 connections (see “Using Auto-Connect” on page 30).

- Also for either role, you can enable the Outbound/Inbound Provisioning option (see “User Provisioning” on page 33).
If you are installing PingFederate and are not sure of your selections, just click Next.

**Note:** If you do not choose a role during installation, you must return to this screen to do so before you can configure connections to partners.

To reach this screen for editing:
1. On the Main Menu under System Settings, click **Server Settings**.
2. Click **Roles and Protocols** under the Server Settings tab.

To choose roles and protocols:
1. Select your federation role(s) and then select at least one protocol.

**Note:** Outbound Provisioning for SaaS applications requires the use of the SAML 2.0. (For more information, refer to the Quick Connection Guide contained in the PingFederate SaaS Connector package for your service provider.)

2. (Optional) If you are using SAML 2.0 and want to configure Auto-Connect, select that feature for your role(s) (see “Using Auto-Connect” on page 30).

**Note:** Clearing this checkbox does not deactivate an existing Auto-Connect configuration in production. If you have already deployed Auto-Connect and wish to suspend the deployment for any reason, use the Initial Setup Summary screens (accessible from the Main Menu) for your respective role.

When you make this selection, two additional steps are added to the System Settings task:
- Metadata Signing (see “Configuring Auto-Connect Metadata Signing” on page 115)
- Metadata Lifetime (see “Configuring Auto-Connect Metadata Lifetime” on page 116)

3. (Optional) If you are using PingFederate as an IdP for provisioning or have installed a SaaS Connector package, select the Outbound Provisioning checkbox.

(For more information, see “Outbound Provisioning for IdPs” on page 34.)

**Note:** After provisioning is configured for a connection, you cannot clear this checkbox—you must delete all provisioning configurations first. To suspend provisioning for an SP partner, you can deactivate the specific configuration (see “Channel Activation and Summary” on page 327). Alternatively, you can deactivate the associated SP connection; note, however, that this will also disable SSO/SLO transactions (see “Editing and Activating a Connection” on page 328).

4. (Optional) If you are using PingFederate as an SP for provisioning, select Inbound Provisioning.

(For more information, see “Just-in-Time Provisioning” on page 35.)
5. (Optional) If you are using SAML 2.0 XASP as an SP for multiple IdP connections, you may select the option to determine dynamically which connection to use, based on the X.509 certificate presented (see “Attribute Requester Mapping” on page 350).

**Tip:** After you make this selection and create XASP IdP connections (see “Configuring the Attribute Query Profile” on page 299), configure dynamic IdP discovery via the Attribute Requester Mapping link on the Main Menu. Once the mapping is configured, you cannot clear the checkbox on the Roles and Protocols screen unless you first delete the mapping.

For general information about XASP, see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started.

6. Click **Next** (or **Save**, if you are modifying existing selections).

For information about configuring settings associated with your selections, see these relevant portions of this manual:

- Chapter 4, “OAuth Configuration”
- Chapter 6, “Identity Provider SSO Configuration”
- Chapter 7, “Service Provider SSO Configuration”
- Chapter 8, “WS-Trust STS Configuration”
- “Configuring IdP Discovery” on page 133

### Specifying Federation Information

This information identifies your federation deployment to your partners, according to the protocol(s) you support.

**Notes:** You must provide an ID that uniquely identifies your federation gateway for each protocol you support. For WS-Trust STS, IDs are required for both SAML 2.0 and SAML 1.x, regardless of browser-based SSO protocol support or the type of token expected to be issued, to ensure that the STS will perform correctly under all conditions.

Each ID normally applies across all connection partners for a given protocol; however, if your implementation requires different IDs for the same protocol, you can use virtual server IDs (see “Federation Server Identification” on page 37).

You can also use a different ID for Auto-Connect transactions (see “Using Auto-Connect” on page 30).
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base URL</strong></td>
<td>The fully qualified host name, port, and path (if applicable) on which the PingFederate server runs. This field is used to populate configuration settings in metadata files (see “Exporting Metadata” on page 57).</td>
</tr>
<tr>
<td><strong>SAML 2.0 Entity ID</strong></td>
<td>This ID defines your organization as the entity operating the server for SAML 2.0 transactions. It is usually defined as an organization’s URL or a DNS address; for example: pingidentity.com. The SAML SourceID used for artifact resolution is derived from this ID using SHA1.</td>
</tr>
<tr>
<td><strong>Auto-Connect Entity ID</strong> (Optional)</td>
<td>(Optional) If you are using Auto-Connect, you can specify a unique ID here for Auto-Connect processing. The value must be a fully qualified URL and should match the CN of your Auto-Connect certificates (see “Auto-Connect Security Model” on page 33). When a value is supplied, this ID is used instead of the SAML 2.0 Entity ID in your server’s Auto-Connect metadata, as well as in associated SSO/SLO requests and responses. Use this field if you have configured regular, static SAML 2.0 connections to other partners and your SAML 2.0 Entity ID is not a fully qualified URL (see “Using Auto-Connect” on page 30).</td>
</tr>
<tr>
<td><strong>SAML 1.x Issuer/Audience</strong></td>
<td>This ID identifies your federation server for SAML 1.x transactions. As with SAML 2.0, it is usually defined as an organization’s URL or a DNS address. The SourceID used for artifact resolution is derived from this ID using SHA1.</td>
</tr>
</tbody>
</table>
### Chapter 3
**System Settings**

To reach this screen:
1. Click **Server Settings** on the Main Menu.
2. Click **Federation Info** under the Server Settings tab.

### Setting System Options

The System Options screen provides global settings that allow you to:

- Turn off automatic multi-connection error checking
- Define a caching interval for data-store validation
- Define HTTP header fields used to identify the originating client IP addresses and the original hostname and port number
- Define whether the incoming proxy terminates HTTPS connections
- Manage system updates

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAML 1.x Source ID</td>
<td>(Optional) If supplied, the Source ID value entered here is used for SAML 1.x, instead of being derived from the SAML 1.x Issuer/Audience.</td>
</tr>
<tr>
<td>WS-Federation Realm</td>
<td>The URI of the realm associated with the PingFederate server. A realm represents a single unit of security administration or trust.</td>
</tr>
</tbody>
</table>

**Note:** The fields available on this screen depend on the federation protocols enabled on your server (see "Choosing Roles and Protocols" on page 106).

To reach this screen:
1. Click **Server Settings** on the Main Menu.
2. Click **Federation Info** under the Server Settings tab.

### Disabling Automatic Connection Validation

Automatic multi-connection error checking occurs by default for all configured connections whenever you access connection lists (Manage Connections screens are available via **Manage All...** links on the Main Menu). The same multi-connection checking also occurs when you access the Manage IdP/SP Adapter Instances screens.
Because validation time increases with the number of connections and adapter instances, this option is provided in case you experience any noticeable delays in loading either of the Manage Connection screens or the adapter management screens. Disabling the feature results in immediate display of the screens, deferring error checking to manual controls on Manage Connection screens.

**Note:** This option does not affect the validation of connections as they are being configured or modified. Also, individual connections are always validated automatically when accessed for editing, regardless of the setting on this screen.

The multi-connection error checking is intended to verify that completed connections have not been affected by any subsequent changes in adapter configurations or other dependencies such as data-store access (see “Data Stores” on page 22).

For more information about connection validation, see:
- “Managing Connection Validation” on page 251 (for SP connections)
- “Managing IdP Connection Validation” on page 357 (for IdP connections)

### Defining Data-Store Validation Intervals

Automatic data-store validation tests occur by default for any adapter-to-adapter mappings or partner connections. This validation test occurs at various points within the administrative console.

The Data Store Validation Interval (secs) field defines the length of time for which a successful data-store validation is cached. (Only successful test results are cached.) The data-store connection is validated using the cached result without performing the test, speeding up the validation process. The default interval is five minutes (300 seconds). A value of 0 turns off the caching and validation tests are executed with each access.

### Defining proxy options

When PingFederate is deployed behind a proxy server or load-balancer, the options described in the following 3 sections enable PingFederate to use information in HTTP headers added by the proxy server to construct correct responses. These options apply globally to all incoming requests.

**Note:** These settings override Jetty proxy options.

#### HTTP Header for Client IP Addresses

The HTTP Header for Client IP Addresses field allows you to globally specify the header name (for example, X-Forwarded-For) where PingFederate should attempt to retrieve the client IP address in all HTTP requests sent to PingFederate. Defining this field helps PingFederate identify the correct client IP address when PingFederate is operating behind a reverse proxy or load balancer.

It is common for proxies to append the IP address from an incoming request to the X-Forwarded-For (or similar) header. If you enter X-Forwarded-For as the HTTP Header for Client IP Addresses, PingFederate combines multiple comma-separated header values into the same order that they are received. Define which IP address you want to use in the list box:
- Leave the default of **Use Last Value** to use the last value in the combined list.
- Select **Use First Value** to use the first value in the combined list.
HTTP Header for Hostname

The HTTP Header for Hostname field allows you to globally specify the header name (for example, X-Forwarded-Host) where PingFederate should attempt to retrieve the hostname and port in all HTTP requests sent to PingFederate. It is common for proxies to append the hostname and port from an incoming request to the X-Forwarded-Host (or similar) header. If you enter X-Forwarded-Host as the HTTP Header for Hostname, PingFederate combines multiple comma-separated header values into the same order that they are received. Define which hostname you want to use in the list box:

- Leave the default of Use Last Value to use the last value in the combined list.
- Select Use First Value to use the first value in the combined list.

Incoming proxy terminates HTTPS connections

The Incoming proxy terminates HTTPS connections field allows you to globally specify that connections to the proxy server are made over HTTPS, even if HTTP is used between the proxy server and PingFederate.

Managing System Updates

The Manage System Updates option on the System Options screen provides online update management for system components. Each component can be configured to use a polling interval to retrieve and install updates automatically. An administrator can also update components manually at any time.

To allow automatic updates at scheduled intervals:

- Select Enable.

To check for an update and install the component immediately:

- Click Update.

To configure polling intervals and other settings as needed:

- Click Manage Update Settings.

Configuring Update Settings

On this screen you can set the polling interval for automatic updates and indicate proxy server settings (if applicable).
To reach this screen:
1. Click **Server Settings** on the Main Menu.
2. Click **System Options** under the Server Settings tab.
3. On the System Options screen, click **Manage System Updates**.
4. On the System Updates screen, click **Manage Update Settings**.

### Configuring Outbound Provisioning Settings

On the Outbound Provisioning screen, you can select the database that PingFederate uses internally to facilitate provisioning for service providers when PingFederate is configured as an IdP (see “Outbound Provisioning for IdPs” on page 34).

This screen is presented only if Outbound Provisioning is enabled for the IdP federation role (see “Choosing Roles and Protocols” on page 106).

**Caution:** A pre-installed, default Hypersonic database is selected for initial setup and testing. However, we strongly recommend that you choose your own, secured database for production deployments.

On this screen, you can also change the provisioning synchronization frequency—that is, how often PingFederate checks the local user store for changes.

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polling Interval (days)</td>
<td>The number of days used to determine how often PingFederate checks for new updates. Valid values range from 1 to 90.</td>
</tr>
<tr>
<td>Proxy Host</td>
<td>The domain name or IP address of the proxy host.</td>
</tr>
<tr>
<td>Proxy Port</td>
<td>The proxy port number. This field is an integer between 1 and 65535.</td>
</tr>
</tbody>
</table>
The database stores the state of synchronization between the source data store and the target data store, enabling periodic checking to determine whether updates are required at the target site. (For information on configuring provisioning as an IdP, see “Configuring Outbound Provisioning” on page 315.)

**Note:** PingFederate has been tested using Hypersonic, Oracle, MySQL, and MS SQL Server databases as internal provisioning data stores. However, any relational database should work as well; adaptable setup scripts used for Hypersonic, Oracle, MySQL, and MS SQL Server are provided in the directory:

```
<pf_install>/pingfederate/server/default/conf/provisioner/sql-scripts
```

**Important:** The certificate CN must match the domain name associated with the Entity ID (see “Specifying Federation Information” on page 109).

### Configuring Auto-Connect Metadata Signing

When Auto-Connect is enabled, PingFederate generates publicly available, signed metadata for partners to use. The metadata contains information about your server configuration (see “Providing Metadata” on page 31).

On the Metadata Signing screen, choose a certificate to use for signing the metadata.

#### Related Information

- **Specifying Federation Information** on page 109
- **Configuring Outbound Provisioning** on page 315
Managing Server Settings

To specify a certificate:

1. Select the certificate from the drop-down list.
   
   If you have not yet created or imported your certificate into PingFederate, click Manage Certificates (see “Digital Signing and Decryption Keys and Certificates” on page 209).

2. (Optional) Select the Signing Algorithm from the drop-down list.
   
   The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

   **Note:** The public certificate is included as part of the metadata and must be trusted by your partner (see “Auto-Connect Security Model” on page 33).

Configuring Auto-Connect Metadata Lifetime

Partners using Auto-Connect metadata will cache it to use for the future requests during the “lifetime” in which the metadata is valid, as configured on the Metadata Lifetime screen. After the metadata lifetime is expired, the metadata is retrieved again.

This metadata expiration ensures that partners always have reasonably up-to-date information about your server. You may elect to use the default time period or change it on the Metadata Lifetime screen.

This screen appears only if Auto-Connect is enabled for either an IdP or SP (see “Choosing Roles and Protocols” on page 106).
Saving and Editing Server Settings

On the Server Settings Summary screen you can view, edit, and save your configuration.

- Click **Save** if you are finished with this configuration, or click any heading to make changes.

Managing Data Stores

PingFederate can connect to local data stores to retrieve user attributes on either the IdP or SP side of an SSO transaction (or both).

**Tip:** Whenever attributes are retrieved from a data store at runtime, PingFederate logs the activity (see "Managing Log Files" on page 44). When you set up access to a data store, you can choose to mask the values of all retrieved attributes in the log files to enhance security and privacy of personal information (see "Attribute Masking" on page 23).

As an IdP, you use this feature whenever you need to fulfill an attribute contract that requires information beyond that which can be derived from the user's session (see “Configuring Attribute Sources and User Lookup” on page 270). For example, this information may include such attributes as an email address, a job title, or any data that can be used to customize a user's experience at the SP site.

As an SP, you can use data stores to retrieve additional attributes to package with the IdP’s assertion data to meet SP adapter requirements (see “SSO Integration Kits and Adapters” on page 14). Such attributes may be needed, for example, to establish authorization levels or to manage the local account.

Either IdP or SP organizations configuring PingFederate for user provisioning must set up connections to data stores (see “User Provisioning” on page 33).

You can add data stores at any time. Standard data stores include JDBC-enabled databases and LDAP v3-compliant directories. Alternatively, you can develop a driver using the PingFederate Custom Source SDK to connect to non-JDBC databases (see the PingFederate SDK Developer’s Guide).

**Note:** You cannot delete or modify a data-store connection if it is associated with an attribute source as part of a partner-connection configuration. You must remove the association first.
To reach this screen:
> Click **Data Stores** on the Main Menu.

**To add a data store:**
1. Click **Add New Data Store**.
2. Select Database, LDAP, or Custom and click **Next**.
3. Continue the configuration:
   - For Database configuration information see “Configuring a JDBC Database Connection” on page 119.
   - For LDAP configuration information see “Configuring an LDAP Connection” on page 122.
   - For Custom configuration information see “Configuring a Custom Data Store” on page 128.
4. Click **Save** when you return to this screen.

**To modify a data store:**
> Click the data store **Description**.

**To delete a data store:**
1. Click **delete** under Action for the data store you want to delete.

**Note:** This option does not appear if the data store is being used in a configuration for provisioning or attribute lookup. To enable deletion, click **Check Usage** to locate the configuration(s) and change the associated data-store dependencies as needed.
2. Click **Save**.

**Configuring a JDBC Database Connection**

You configure access to a database by providing basic JDBC information.

**Note:** Ensure that your database driver JAR file is installed in the `pingfederate/server/default/lib` directory and that the driver is compatible with the JDK version in use and with the target database. You must restart the server after installing the driver.
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>JDBC URL</strong></td>
<td>(Required) The location of the JDBC database, in the format: jdbc:mysql://databaseservername/databasename or jdbc:sqlserver://databaseservername/databasename or jdbc:oracle:thin://databaseservername/databasename where databaseservername is the DNS host name (or IP) of the server hosting the database, and databasename is the name of a database on that server. Note: For MySQL, to enable automatic reconnection attempts if the connection is not available at runtime, enter a SQL statement in the Validate Connection SQL field below and add the following query string to the JDBC URL: ?autoReconnect=true For more information, see the field description for Validate Connection SQL below.</td>
</tr>
<tr>
<td><strong>Driver Class</strong></td>
<td>(Required) The name of the driver class used to communicate with the source database. For example, com.mysql.jdbc.Driver, com.microsoft.sqlserver.jdbc.SQLServerDriver, or oracle.jdbc.OracleDriver. This class should be supplied by the database software vendor in a JAR file, which must be present in the pingfederate/server/default/lib directory.</td>
</tr>
<tr>
<td><strong>Username</strong></td>
<td>(Required) The name that identifies the user when connecting to the database.</td>
</tr>
</tbody>
</table>
Managing Data Stores

To reach this screen for editing:
1. Click Data Stores on the Main Menu.
2. Click the data-store Description link on the Manage Data Stores screen.

To configure a new data store:
1. Click Data Stores on the Main Menu.
2. Click Add New Data Store.
3. Select Database and click Next.

To establish access to a database:
1. Enter the applicable JDBC URL.
   This URL is used to identify the data store in lists. Example: jdbc:mysql://10.0.1.81:3306/idp
2. Enter the Driver Class.
   Example: com.mysql.jdbc.Driver

   Note: The driver JAR file must be loaded into the directory:
   pingfederate/server/default/lib
3. Enter a valid Username and Password.
4. (Optional) Enter a valid SQL statement in the Validate Connection SQL field.
   For information, see the Description of this field in the “Field Descriptions” table above.

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>The password needed to access the database.</td>
</tr>
<tr>
<td>Validate Connection SQL</td>
<td>(Optional, but recommended) A simple SQL statement used by PingFederate at runtime to verify that the database connection is still active and to reconnect if needed. If a SQL statement is not provided here, PingFederate may not be able to reconnect to the database if the connection is broken. Note: To use this feature for MySQL, you must also add a query parameter to the JDBC URL (see information in field description above). Important: Ensure that the SQL statement is valid for your database. Examples: (For Oracle or MySQL) SELECT 1 from dual (For SQL Server) SELECT getdate()</td>
</tr>
<tr>
<td>Mask Values in Log (Checkbox)</td>
<td>Determines whether all attribute values returned from this data store will be masked in PingFederate log files (see “Attribute Masking” on page 23).</td>
</tr>
<tr>
<td>Allow Multi-Value Attributes (Checkbox)</td>
<td>When selected (the default), indicates that this JDBC data store can select more than one record from a column and return the results as a multi-value attribute. Otherwise, a query returns only the first value in the column.</td>
</tr>
</tbody>
</table>
5. (Optional) Select Mask Values in Log. For information, see “Attribute Masking” on page 23.

6. (Optional) Click Advanced. Use this option to change default sizes or look-up time-outs, or to validate the connection using a specific SQL call (see “Setting Advanced Options” on page 121).

7. Click Next.

**Note:** PingFederate tries to connect to the database at this point. If it cannot, there may be a problem with your settings.

8. On the Summary screen, click Done.

9. Click Save on the Manage Data Stores screen.

**Setting Advanced Options**

Use the Advanced Database Options screen to change default pool sizes or look-up time-outs, or to validate the connection using a specific SQL call.

<table>
<thead>
<tr>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
</tr>
<tr>
<td>Minimum Pool Size</td>
</tr>
<tr>
<td>Maximum Pool Size</td>
</tr>
<tr>
<td>Blocking Timeout (ms)</td>
</tr>
<tr>
<td>Idle Timeout (min)</td>
</tr>
</tbody>
</table>

**To reach this screen for editing:**

1. Click Data Stores on the Main Menu.
2. Click the Data Store Description link on the Manage Data Stores screen.
3. Click the Advanced button on the Database Config screen.
To configure a new data store:
1. Click Data Stores on the Main Menu.
2. Click Add New Data Store.
3. Select Database and click Next.
4. Enter information on the Database Config screen and click the Advanced button.

   Internally, PingFederate is preconfigured to use published Jetty server default values. To view or restore these values, click Apply Defaults.

Configuring an LDAP Connection

This screen establishes a connection between the PingFederate server and an LDAP data store.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname(s)</td>
<td>The DNS name or IP address of the data store, which may include a port number; example: 181.20.42.130:389. For failover, you can enter one or more backup LDAP servers, each separated by a space. <strong>Note:</strong> If more than one Hostname is entered, each server must be accessible using the same User DN and Password (or via Bind Anonymously).</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAP Type</td>
<td>If you are using this data store for Outbound Provisioning and your LDAP store is either Active Directory (AD) or Oracle Directory Server, select the applicable Type (see “Outbound Provisioning for IdPs” on page 34). Identifying the LDAP type permits PingFederate to configure many provisioning settings automatically (see “Modifying Source Settings” on page 320). The LDAP type is also used to enable password-change messaging between AD and PingFederate (see “HTML Form Adapter Configuration” on page 515). <strong>Tip:</strong> If you are using Outbound Provisioning and your LDAP server is not AD or Oracle, you may wish to define a custom LDAP Type to streamline the provisioning configuration (see “Defining an LDAP Type” on page 124).</td>
</tr>
<tr>
<td>Bind Anonymously (Checkbox)</td>
<td>(Optional) Username and password are not required (see procedure below).</td>
</tr>
<tr>
<td>User DN</td>
<td>The username credential required to access the data store. <strong>Important:</strong> The user must have permission to search the directory for user-account information. For security, this user should have read-only access.</td>
</tr>
<tr>
<td>Password</td>
<td>The password credential required to access the data store.</td>
</tr>
<tr>
<td>Use SSL (Checkbox)</td>
<td>(Optional) Connects to the LDAP data store using secure SSL/TLS encryption (LDAPS). This selection applies equally to any secondary servers specified in Hostname(s). <strong>Caution:</strong> We recommend that all LDAP connections be secured using LDAPS.</td>
</tr>
<tr>
<td>Mask Values in Log (Checkbox)</td>
<td>(Optional) Determines whether all attribute values returned from this data store will be masked in PingFederate log files (see “Attribute Masking” on page 23).</td>
</tr>
</tbody>
</table>

**To reach this screen for editing:**

1. Click **Data Stores** on the Main Menu.
2. Click the data-store Description link on the Manage Data Stores screen.
3. Click the LDAP Configuration link on the Summary Info screen.
To establish a connection to an LDAP data store:

1. Enter the applicable Hostname(s).
   Hostnames identify this LDAP configuration in selection lists elsewhere in the administrative console.
2. (Optional) Select an LDAP Type from the drop-down list.
3. Either:
   • Select Bind Anonymously if your LDAP interface supports anonymous binding and if no credentials are needed to access the data store.
   Or:
   • Enter a valid User DN and Password.

4. (Optional) Select Use SSL.
5. (Optional) Select Mask Values in Log.
6. (Optional) Click Advanced.
   Use this option to configure LDAP pooling properties (see “Setting Advanced LDAP Options” on page 125).
7. Click Next.

8. On the Summary screen, click Done.
9. Click Save on the Manage Data Stores screen.

Defining an LDAP Type

If you are using Outbound Provisioning and your user-management LDAP server is not Active Directory or the Oracle Directory Server, you can define a custom LDAP Type for PingFederate to use to streamline the provisioning configuration (see “Outbound Provisioning for IdPs” on page 34).

When the LDAP server is defined, its type appears in the LDAP Type drop-down list on the LDAP Configuration screen (see previous section). When the data store is selected as the source for provisioning, a number of other settings can be automatically configured (see “Modifying Source Settings” on page 320).

To define an LDAP Type:
1. If you are using the LDAP Configuration screen, click Previous or Cancel.
2. Copy and rename the file sample.template.txt:
   <pf_install>/pingfederate/server/default/conf/template/ldap-templates

Note: PingFederate attempts to connect to the data store at this point. If it cannot, there may be a problem with your settings.

Note: The fields referenced in the following steps are detailed in the Field Descriptions table above.

Note: If you choose an anonymous binding, ensure that this access level provides permission to search the directory for user-account information.
3. Change the `template.name` in the new template file.
   The `template.name` you specify appears in the LDAP Type list on the LDAP Configuration screen when you save the template.

4. Modify other property values in the file to match the corresponding configuration of your LDAP server.
   The properties are used in the Outbound Provisioning setup (see “Modifying Source Settings” on page 320).

5. Save the new template file.

### Setting Advanced LDAP Options

PingFederate maintains a search pool and a bind pool for each LDAP data store instance for optimal performance. The search pool is meant for LDAP directory searches. The bind pool is meant for LDAP bind authentication purposes.

Use the Advanced LDAP Options screen to change default pool settings for the related data store. These settings are applicable to both the search pool and the bind pool.

#### Field Descriptions

**Tip:** The defaults on this screen are conservative based on the default thread-pooling limits allowed by the installed PingFederate Web container. (These limits are under “Server Thread Pool” in `jetty-runtime.xml` located in `<pf_install>pingfederate/etc`.)

If any changes are made to thread pooling, we recommend updating settings as outlined in the following table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Connection on Borrow (Checkbox)</td>
<td>Indicates whether objects are validated before being borrowed from the pool.</td>
</tr>
</tbody>
</table>
### Field | Description
--- | ---
Test Connection on Return (Checkbox) | Indicates whether objects are validated before being returned to the pool.

Create New Connection If Necessary (Checkbox) | Indicates whether temporary connections can be created when the Maximum Connections threshold is reached. Temporary connections are managed automatically.  
**Note:** If disabled, when the Maximum Connections threshold is reached, subsequent requests relying on this LDAP data store instance may fail.

Minimum Connections | The smallest number of connections that can remain in each pool, without creating extra ones. A minimum value of 1 creates one connection in the search pool and one connection in the bind pool.  
**Note:** For optimal performance, the value for this setting should be equal to 50% of the `maxThreads` value in the Jetty server configuration (see the Tip above this table).

Maximum Connections | The largest number of active connections that can remain in each pool without releasing extra ones. The value must be greater than or equal to the Minimum Connections value.  
**Note:** For optimal performance, the value for this setting should be equal to 75% to 100% of `maxThreads` value in the Jetty server configuration (see the Tip above this table).

Maximum Wait (Milli) | The maximum number of milliseconds the pool waits for a connection to become available when trying to obtain a connection from the pool. A value of -1 causes the pool not to wait at all and to either create a new connection or produce an error (when no connections are available).

Time Between Eviction (Milli) | The frequency, in milliseconds, that the evictor cleans up the connections in the pool. A value of -1 disables the evictor.

Read Timeout (Milli) | The maximum number of milliseconds a connection waits for a response to be returned before producing an error. A value of -1 causes the connection to wait indefinitely.

Connection Timeout (Milli) | The maximum number of milliseconds that a connection attempt should be allowed to continue before returning an error. A value of -1 causes the pool to wait indefinitely.

---

**To reach this screen for editing:**

1. Click **Data Stores** on the Main Menu.
2. Click the data-store Description link on the Manage Data Stores screen.
3. Click the LDAP Configuration link on the Summary Info screen.
Chapter 3  
System Settings

4. Click Advanced.

Specifying LDAP Binary Attributes

The LDAP Binary Attributes screen allows you to specify which attributes must be handled as binary data for use in attribute contract fulfillment. Binary data cannot be used in an assertion. Encoding must be applied and is handled on a connection basis when binary attributes are selected for attribute mapping.

To reach this screen for editing:
1. Click Data Stores on the Main Menu.
2. Click the data-store Description link on the Manage Data Stores screen.
3. Click the LDAP Configuration link on the Summary Info screen.
4. Click Advanced.
5. Click LDAP Binary Attributes.

To configure LDAP Binary Attributes:
1. Enter the attribute name in the text box.
2. Click Add.
3. Repeat steps 1 and 2 for other attributes as needed.
4. Click Done.

To edit LDAP Binary Attributes:
1. Click Edit under Action for the binary attribute.
2. Make your change and click Update.

To delete an LDAP Binary Attribute
1. Click Delete under Action for the binary attribute.

Configuring a Custom Data Store

Developers can use the PingFederate Custom Source SDK to create specific drivers for non-JDBC/LDAP data stores (or more sophisticated JDBC/LDAP lookups) including, for example, flat files or SOAP-connected databases (see the PingFederate SDK Developer’s Guide).

Once the data-store driver is installed, you can select it on the Custom Data Store Type page.
Managing Data Stores

To start configuring a Custom Data Store:

1. Enter a unique Instance Name.
   You can create more than one instance of the same Data Store Type for use with different connection partners, as needed.

2. Select the Data Store Type.

3. (Optional) Select Mask Values in Log.
   For information see “Attribute Masking” on page 23.

4. Click Next.

Configuring a Custom Data Store Instance

This screen will vary depending on the implementation. Below is a sample for a SOAP-enabled database driver. The screen shown below is only an example of a custom data store and is not available in the PingFederate distribution.

To configure the driver instance for use with a partner connection:

- Enter or select required information and click Next.

Adapter Actions

Custom data store adapters may be written to interface PingFederate to perform configuration assistance or validation actions (for example, testing a connection to a database). Actions may also include generation of parameters that might need to be set manually in a configuration file.

- To invoke an adapter action (when applicable), click its link on the Adapter Actions screen.

Editing and Saving a Data Store

On the Data Store Summary page, you can view or edit your configuration.
To modify the configuration:
▶ Click the heading above the information you want to change.

To save a new configuration:
▶ Click Done on the Summary screen and then Save on the Manage Data Stores screen.

Defining an Account-Linking Data Store

When an SP is configured to use account linking for an IdP connection, PingFederate uses an embedded Hypersonic database as the account-link repository (see “Account Linking” on page 17). This default implementation does not require any changes to PingFederate to support account linking. However, you can manually customize PingFederate to store account links in a different data store—either a different database or an LDAP directory. You might want to do this for any of several reasons, including:

- You are running a cluster of PingFederate runtime engines (see the PingFederate Server Clustering Guide). This scenario requires that you use an external database or directory for account links to ensure proper local user lookup.
- You have performance or scalability requirements that exceed the Hypersonic database’s capabilities.
- You and your federation partner previously established a different system for creating and mapping opaque pseudonyms, and PingFederate needs access to the system.

Changing the Account-linking Database

Changing the default database involves creating a table in your JDBC database to support account linking, and modifying PingFederate configuration XML files to use the database.

To create a database table for account linking:
▶ Run one of the table-setup scripts provided in the directory:

```
<pf_install>server\default\conf\account-linking\sql-scripts
```

If a script is not provided for your database, you can derive the setup from information available in any of the other scripts.

To change the account-linking database:

1. If you have not already done so, create a connection to the database you want to use (see “Configuring a JDBC Database Connection” on page 119).
   Be sure to save the configuration on the Manage Data Stores screen.
2. Any time after saving the database connection, return to the Manage Data Stores screen from the Main Menu.
   (To reach the screen, click Data Stores under System Settings.)
3. Copy the System ID for the database you want.
4. In the directory `<pf_install>/server/default/data/config-store`, open the file:
   Between the XML tags for the item named PingFederateDSJNDIName, insert the System ID you copied at Step 3 and save the file.
5. Start or restart PingFederate.

**Note:** If you are running PingFederate in a cluster, push the new configuration to other server nodes. For more information, see the PingFederate Server Clustering Guide.

### Changing the Default Data Store to use LDAP

Changing the default data store to use LDAP involves modifying PingFederate configuration XML files to use the LDAP directory.

**To use an LDAP directory:**

1. If you have not already done so, create a connection to the LDAP data store you want to use (see “Configuring an LDAP Connection” on page 122).
   
   Be sure to save the configuration on the Manage Data Stores screen.

2. Any time after saving the LDAP data store connection, return to the Manage Data Stores screen from the Main Menu.
   
   (To reach the screen, click Data Stores under System Settings.)

3. Copy the System ID for the data store you want.

4. In the directory `<pf_install>/server/default/conf/META-INF`, open the file:
   
   `hivemodule.xml`
   
   Locate the Service-Point ID for AccountLinkingService and change the value of the create-instance class to:
   
   `org.sourceid.saml20.service.impl.AccountLinkingServiceLDAPImpl`

5. In the directory `<pf_install>/server/default/data/config-store`, open the file:
   
   `org.sourceid.saml20.service.impl.AccountLinkingServiceLDAPImpl.xml`
   
   Insert the following values between the XML tags for the these items:
   
   - PingFederateDSJNDIName: System ID you copied at Step 3
   - UserSearchBase: LDAP location where searches begin—for example, CN=Users,DC=LDAPDir,DC=com
   - UsernameAttribute: LDAP attribute that represents the user identifier—for example, Active Directory is sAMAccountName
   - AccountLinkDataAttribute: LDAP attribute used to store account linking data

   **Note:** The AccountLinkDataAttribute can be any multi-valued string attribute on a user object class. We recommend that you extend the LDAP schema with a custom attribute for use here. See the MSDN article for further information on extending the Active Directory schema (http://msdn.microsoft.com/en-us/library/ms676900%28v=VS.85%29.aspx).

6. Start or restart PingFederate.
7. If you are running PingFederate in a cluster, push the new configuration to other server nodes (see the PingFederate Server Clustering Guide).

**Important:** You must manually apply the changes made in Step 4 to the hivemodule.xml file on each server node in a cluster and then start or restart PingFederate.

**Note:** User accounts to be linked must exist in the LDAP directory prior to establishing the account link. The Account Linking service does not add users to the LDAP data store but simply updates AccountLinkDataAttribute for a given user.

### Defining an OAuth Grant Data Store

As is the case for Account Linking (see “Defining an Account-Linking Data Store” on page 129), PingFederate uses its internal Hypersonic database by default to maintain persistent access-token grants for the OAuth AS (see “Persistent vs. Transient Grants” on page 12). In a production environment, an administrator should change this configuration to use an external enterprise database.

**Important:** For server clustering, an external database is required.

Changing the default data store for OAuth persistent grant storage involves modifying a PingFederate configuration XML file to point to the appropriate database.

**To create the database table for grant storage:**

- Run one of the table-setup scripts provided in the directory:
  `<pf_install>/server/default/conf/access-grant/sql-scripts`

  If a script is not provided for your database, you can derive the setup from information available in any of the other scripts.

**To change the database:**

1. If you have not already done so, create a connection to the database you want to use (see “Configuring a JDBC Database Connection” on page 119).

   Be sure to save the configuration on the Manage Data Stores screen.

2. Any time after saving the database connection, return to the Manage Data Stores screen from the Main Menu.

   (To reach the screen, click Data Stores under System Settings.)

3. Copy the System ID for the database you want.

4. In the directory `<pf_install>/server/default/data/config-store`, open the file:

   `org.sourceforge.oauth20.token.AccessGrantManagerJdbcImpl.xml`

   Change the value of the item named `PingFederateDSJNDIName` to the System ID you copied in the previous step and save the file.
Managing Data Stores

5. Start or restart PingFederate.

Note: If you are running PingFederate in a cluster, push the new configuration to other server nodes. For more information see the PingFederate Server Clustering Guide and restart PingFederate.

Defining an OAuth Client Data Store

PingFederate can be configured to use a centralized database for OAuth client records, allowing the records to be managed via either the administrative console (see “Client Management” on page 177) or a built-in OAuth Client Web Service (see “OAuth Client Management Service” on page 560). Database management is recommended for scalability in deployments where large numbers of clients are anticipated.

As with the OAuth grant database (see previous section), when database usage is enabled, PingFederate uses its internal Hypersonic database by default.

Important: In a production environment, an administrator should change the default configuration to use an external enterprise database. For server clustering, an external database is required, since the Hypersonic database cannot be shared across other PingFederate instances.

Enabling database usage and changing the default database involve modifying two PingFederate configuration files and restarting the server. For production databases, an administrator must also create a database table for OAuth clients using an SQL script.

To enable usage of an OAuth-client database:

1. In the directory `<pf_install>/server/default/conf/META-INF`, open the file: hivemodule.xml
2. Search the file for: ClientManagerXmlFileImpl
   Replace with: ClientManagerJdbcImpl
   You can reverse the setting if you ever want to revert to the default XML file storage implementation.

Caution: If clients are already defined using the default XML implementation, the records are not moved to a database: they must be recreated. Likewise, if the configuration is changed back to the default, database records are not ported to XML files.

3. Restart PingFederate.
4. If you are running PingFederate in a server cluster, repeat the previous steps for each server node.
5. If you are ready to define an external enterprise database, follow the steps below.
To change the database:

1. For the database you want to use, run one of the table-setup scripts provided in the directory:

   `<pf_install>server\default\conf\oauth-client-management\sql-scripts`

   If a script is not provided for your database, you can derive the setup from information available in any of the other scripts.

2. If you have not already done so, create a connection to the database (see “Configuring a JDBC Database Connection” on page 119).

   Be sure to save the configuration on the Manage Data Stores screen.

3. Any time after saving the database connection, return to the Manage Data Stores screen from the Main Menu.

   (To reach the screen, click Data Stores under System Settings.)

4. Copy the System ID for the database you want.

5. In the directory `<pf_install>/server/default/data/config-store`, open the file:

   `org.sourceid.oauth20.domain.ClientManagerJdbcImpl.xml`

   Change the value of the item named `PingFederateDSJNDINName` to the System ID you copied in the previous step and save the file.

   **Note:** If you are running PingFederate in a cluster, be sure to make this modification on the administrative server.

6. Start or restart PingFederate.

   **Note:** If you are running PingFederate in a cluster, push the new configuration to other server nodes (see the PingFederate Server Clustering Guide).

### Configuring IdP Discovery

PingFederate provides two kinds of IdP discovery:

- SAML 2.0 standard IdP Discovery (see “IdP Discovery” in the “Supported Standards” chapter of Getting Started)

- Proprietary Idp discovery using a persistent cookie written by an SP PingFederate server

Standard IdP Discovery is configured in the administrative console (see the next section).

Discovery based on a PingFederate proprietary cookie is configured in an XML file (see “IdP Discovery Using a Persistent Cookie” on page 136). Note that this method can be used in conjunction with any of the federation standards.

### Standard IdP Discovery

SAML IdP Discovery provides a cookie-based look-up mechanism used to identify a user’s IdP dynamically during an SP-initiated SSO event, when the IdP is not otherwise specified. To enable this feature, IdP Discovery must be selected on the Roles and Protocols screen in the System Settings configuration (see “Choosing
Roles and Protocols” on page 106). Then click **IdP Discovery** under System Settings on the Main Menu to reach this screen.

To continue, click **Configure IdP Discovery**.

### Choosing Domain Cookie Settings

On the Domain Cookie Settings screen, you choose the discovery role or roles that PingFederate will play.

The choices that appear on this screen depend on whether PingFederate is acting as an SP, an IdP, or both; or as an IdP Discovery server only (see “Choosing Roles and Protocols” on page 106).

**To reach this screen:**

1. Click **IdP Discovery** under System Settings on the Main Menu.
   
   If this link is not available, then IdP Discovery is not yet enabled (see “Choosing Roles and Protocols” on page 106).

2. On the IdP Discovery screen, click **Configure IdP Discovery**.

   For a detailed discussion of selections on this screen, see “IdP Discovery” in the “Supported Standards” chapter of Getting Started.

### Configuring a Common Domain Service

A Common Domain Service is where PingFederate reads and/or writes authentication information contained in shared cookies, as determined by whether your site is an SP or IdP, respectively. (The service is shared if your PingFederate server is acting in both roles.)
To reach this screen:
1. Click IdP Discovery under System Settings on the Main Menu.
   If this link is not available, then IdP Discovery is not yet enabled (see “Choosing Roles and Protocols” on page 106).
2. On the IdP Discovery screen, click Configure IdP Discovery.
3. Click Common Domain Service under the Configure IdP Discovery tab.
   This step is not available if your server is configured for IdP Discovery only (see “Choosing Roles and Protocols” on page 106).

To configure the Common Domain Service:
1. Enter the Base URL.
   You must use SSL/TLS (HTTPS) for a common domain.
2. Enter and confirm a Pass phrase that a Web application must use to access the domain.

Configuring a Local Common Domain Server

A Local Common Domain Server is where PingFederate reads (as an SP) or writes (as an IdP) a common domain cookie (CDC) for IdP Discovery.

To reach this screen:
1. Click IdP Discovery under System Settings on the Main Menu.
   If this link is not available, then IdP Discovery is not yet enabled (see “Choosing Roles and Protocols” on page 106).
2. On the IdP Discovery screen, click Configure IdP Discovery.
3. Click Local Common Domain Server under the Configure IdP Discovery tab.
   This step is available only if the common-server option is selected under Domain Cookie Settings (see “Configuring IdP Discovery” on page 133).
To configure the Local Common Domain Server:

1. Enter the Common Domain.
   Your entry must include an initial period ( . ); for example:
   .pingidentity.com
2. Enter the Cookie Lifetime.
   The range is 1 to 1,825 days; or to indicate a nonpersistent, session cookie,
   enter -1.
3. Enter and confirm a Pass phrase that a Web application must use to access the
domain.


Important: This configuration does not automatically enable the setting of
the cookie itself at runtime. Follow one of the two options described
below.

To enable setting the common cookie at runtime:

Either:

Ensure that, prior to launching any SSO events, the Web application that
implements IdP Discovery sets the cookie using the PingFederate endpoint
intended for that purpose (see “/idp/writecdc.ping” on page 532).

Or:

Enable setting the cookie at runtime during SSO events by enabling IdP
Discovery for the desired SP connection on the Connection Options page (see
“Choosing Connection Options” on page 254).

Editing and Saving the Configuration

After configuring or modifying IdP Discovery settings, you can review the
configuration on the Summary screen.

If you are finished with the configuration, click Save; otherwise, click any
heading to make changes.

IdP Discovery Using a Persistent Cookie

PingFederate's proprietary IdP-discovery method makes use of an IdP Persistent
Reference Cookie (IPRC) to track the identity provider with whom a user last
authenticated. There are three significant differences between standard IdP
Discovery and the IPRC method:

- Standard IdP Discovery may be used only with SAML 2.0; the IPRC may be
  used with any federation protocol.
- The Common Domain Cookie (CDC) may be configured as a temporary,
  session-based cookie; the IPRC always persists for a configurable period of time.
- The CDC is set by the IdP and readable by both federation partners; the IPRC is
  set by the SP, using information in the SAML assertion, and cannot be accessed
  by the IdP.

Configuration

Enable the IPRC feature for your SP site using the configuration file
org.sourceforge.websso.profiles.sp.IdpIdCookieSupport.xml located in
the directory <pf_install>/pingfederate/server/default/data/config-
store.
Note that the deployed connection configuration between SP and IdP partners must include SP-initiated SSO (see “Configuring SAML Protocol Settings” on page 392).

To enable IPRC:
1. In the XML configuration file cited above, set the value of EnableIdpIdCookie to true.
2. (Optional) Change the default value(s) of any of the remaining elements in the configuration, as described in the following table:

<table>
<thead>
<tr>
<th>IdpIdCookieName</th>
<th>The name of the IPRC set by the SP installation (default: IdPId). Note that the cookie name cannot contain any of the following characters: &amp;, &gt;, &lt;, comma, semicolon, space.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IdpIdCookieLifeTimeInDays</td>
<td>The lifetime for the cookie (default: 365 days, maximum: 24,855 days). The browser will delete the cookie when the period is expired.</td>
</tr>
<tr>
<td>ShowIdpSelectionList</td>
<td>If set to true (the default), the SP displays a list of IdPs that can be used to initiate the SSO event if the cookie is not set. If set to false, the SP installation generates an error page.</td>
</tr>
</tbody>
</table>

3. Start or restart PingFederate.

Note: Once an IPRC cookie is set, the only way to change the IdP to whom the SP will send Authentication Requests for the user is to do one of the following: wait for the cookie to expire, delete the cookie, or perform IdP-initiated SSO using the new IdP.

IdP-to-SP Adapter Mapping

This configuration is provided for special use cases in which PingFederate is acting as both an IdP and an SP, and user attributes from an IdP adapter are used to create an authenticated session via an SP adapter on the same PingFederate server. Generally, these cases involve SaaS providers who may not support standards-based SSO but do provide proprietary SSO with “delegated authentication” (for example, Salesforce and Workday).

The mapping may also be used to enable the Google Apps Password Manager (available separately with the Google Apps Connector—see “Outbound Provisioning for IdPs” on page 34).

In effect, this configuration provides an alternative to setting up complete connections to send SAML assertions and other messages back and forth between an IdP and an SP running on the same PingFederate server (a loopback configuration) to enable nonstandard use cases. Instead, attributes that would normally be sent in an assertion are mapped directly from the IdP authentication adapter to an SP adapter, resulting in a secure SP user session.

To use this configuration, ensure that you have already configured the required IdP and SP adapter instances. Note that you may reuse instances that are also in use for connection configurations. For more information, see:
Managing Mappings

On the Manage Mappings screen you can add, modify, or delete IdP-to-SP Adapter Mappings.

To add a mapping:

- Select the adapter Source Instance (IdP) and Target Instance (SP) and click Add Mapping.

**Note:** You can create only one mapping of a source to the same target. However, you can map different sources to the same target, and vice versa.

To edit a mapping:

- Click the mapping name.

To delete a mapping:

- Click Delete under Actions for the mapping and then click Save.

Assigning a License Group

Adapter-to-adapter mapping is considered a connection for licensing purposes. If your PingFederate license manages connections by groups, select a license group for this mapping configuration.

**Note:** This screen is not displayed for unrestricted or other types of licenses.

To assign a License Group:

- Select the License Group from the drop-down list and click Next.
Configuring Attribute Lookup for IdP-to-SP Adapter Mapping

Attribute sources are specific data store or directory locations containing information that may be required to fulfill the SP adapter contract.

This portion of the IdP-to-SP adapter configuration allows you to configure one or more data stores to look up attributes and to set up search parameters.

- If data-store lookup is not required, click Next.
- If data-store lookup is required, click Add Attribute Source and complete the setup steps (see “Choosing a Data Store (Optional)” next).

To modify an attribute source configuration:
1. Click the attribute source Description link.
2. Click Save on the screen you change.

Note: Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click Save or Done when either of those options appears.

Choosing a Data Store (Optional)

Under some circumstances, to fulfill the SP adapter contract, you may need to look up user attributes in data stores to supplement those available from the user's session via the IdP adapter. This screen provides that option.

To define an attribute source:
1. Enter an Attribute Source Id to uniquely identify the data source for the mapping.
2. Use Attribute Source Description to specify an attribute source name that distinguishes this user lookup for the selected data store.

**Note:** PingFederate appends this description to the data store type in the Source list on the Adapter Contract Fulfillment screen (see “Adapter Contract Fulfillment” on page 140).

3. Choose an Active Data Store and click **Next**.

A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click **Manage Data Stores** to add it (see “Managing Data Stores” on page 117).

**Note:** Data-store lookup screens for this configuration are functionally identical to those used for an IdP connection. Refer to the table below to find applicable sections in this manual containing configuration information and procedures. Then continue to the next section, “Adapter Contract Fulfillment”.

- For information about data-store lookup configuration screens, depending on the type of data store, use this table:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Manual Section</th>
</tr>
</thead>
</table>
| JDBC            | • “Selecting a JDBC Database Table and Columns” on page 273  
                  | • “Configuring a Database Filter (WHERE Clause)” on page 275  |
| LDAP            | • “Configuring an LDAP Directory Search” on page 277  
                  | • “Configuring an LDAP Filter” on page 280  |
| Custom          | • “Configuring Custom Source Filters” on page 282  
                  | • “Selecting Custom Source Fields” on page 282  |

**Adapter Contract Fulfillment**

The next step in this configuration is to map values from the IdP adapter into the attributes required by the SP adapter (the Adapter Contract).
Map each attribute to fulfill the Adapter Contract from one of these Sources:

- **Adapter**
  When you make this selection, the associated Value drop-down list is populated by the IdP adapter.

- **Context**
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **LDAP/JDBC/Custom (if a data store is configured)**
  Values are returned from a data source. When you make this selection, the Value list is populated by the LDAP, JDBC, or Custom attributes you identified.

  **Note:** PingFederate appends a description in parentheses for configured data store lookups (see “Configuring Attribute Lookup for IdP-to-SP Adapter Mapping” on page 139).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the IdP Adapter, using the \$\{attribute\} syntax.

  You can also enter values from your data store, when applicable, using this syntax:

  \$\{ds.attr-source-id.attribute\}

  where attr-source-id is the Attribute Source Id value (see “Choosing a Data Store (Optional)” on page 139) and attribute is any of the data store attributes you select.

**To map attributes:**

1. Choose a **Source** for each SP Adapter Contract attribute.
   See the list under “Map each attribute to fulfill the Adapter Contract from one of these Sources:” above.

2. Choose (or enter) a **Value** for each attribute.
   All values must be mapped.

3. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).
Configuring a Default Target URL (Optional)

Use this screen to assign a default target URL for this adapter-to-adapter mapping. Entering a URL in the Default Target URL field overrides any SP Default URL SSO setting (see “Configuring Default URLs” on page 346).

**Note:** If specified, the adapter-to-adapter endpoint parameter `TargetResource` overrides any default target URL for an adapter-to-adapter mapping (see “System-Services Endpoints” on page 540).

![Default Target URL Configuration](image)

Configuring Issuance Criteria (Optional)

Use this screen to define criteria PingFederate can evaluate to determine whether to issue an SP adapter token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can SSO to protected resources.

![Issuance Criteria Configuration](image)

**To configure Issuance Criteria:**

1. Select a source containing attributes for token authorization.

   Associated attributes appear in the Attribute Name drop-down list:
   - Adapter—Select to access attributes from the IdP Adapter.
   - Context—Select to use values returned from the context of the transaction at runtime.

   **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.
Chapter 3
System Settings

- LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.

**Note:** PingFederate appends a description in parentheses for configured data store lookups (see “Configuring Attribute Lookup for IdP-to-SP Adapter Mapping” on page 139).

- Mapped Attributes – Select to access the required attributes from the adapter.

2. Select an attribute name.
3. Select the Condition you want to apply.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

Error results are handled in one of two ways:
- **Redirect** – When an InErrorResource URL is provided, the value of the Error Result field is used by an ErrorDetail query parameter in the redirect URL.
- **Template** – When an InErrorResource URL is not provided, the value of the Error Result field is used by the variable $errorDetail in the idp.sso.error.page.template.html template (for more information on this and other user-facing templates, see “Customizing User-Facing Screens” on page 87).

**Note:** Using an error code in the Error Result field allows the error template or a Web application to process the error result in a variety of ways—for example, a redirect to another page or e-mail to an administrator.

If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click **Add**.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the **Show Advanced Criteria** button to appear (see “Enabling and Disabling Expressions” on page 573).
Important: When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

- Use the in-line editor box to enter the OGNL expression.
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click Add. Continue to add expressions, as needed.
- Click the Test link to input values and test the resulting output for the expression.

Note: For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click Next (or Done, if you are finished with the configuration, and then Save on the Manage Mappings screen).

To edit Issuance Criteria:
  - Click Edit under Actions for the criteria.

To delete Issuance Criteria:
  - Click Delete under Actions for the criteria and then click Save.

Using the Summary Screen

When you have finished configuring IdP-to-SP Adapter Mapping, you can review the configuration on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click Done and then Save on the Manage Mappings screen.

STS Token Exchange Mapping

This configuration is provided for use cases in which the PingFederate WS-Trust STS exchanges one type of security token for another without requiring a SAML token to be generated in between (see “About WS-Trust STS” on page 6). Use this configuration, for example, to convert a user's Kerberos token to a third-party proprietary WAM session token.

In effect, this configuration provides an alternative to setting up complete STS connections to make such an exchange using the same instance of PingFederate.
Instead, incoming user attributes from an IdP token processor are mapped directly to an SP token generator.

To use this configuration, ensure that you have enabled both the IdP and SP roles for PingFederate, including the WS-Trust protocol (see “Enabling the WS-Trust STS” on page 437). Also, be sure to configure the required token-translator instances. Note that you may reuse instances that are also in use for STS connection configurations. For more information, see:

- “Configuring Token Processors” on page 443
- “Configuring Token Generators” on page 476

Managing Token Mappings

On the Manage Mappings screen you can add, modify, or delete token-to-token mappings.

To reach this screen:

- Click STS Token Exchange Mapping under System Settings on the Main Menu.

To add a mapping:

- Select the token-processor Source Instance and the token-generator Target Instance and click Add Mapping.

**Note:** You can create only one mapping of a source to the same target. However, you can map different sources to the same target, and vice versa.

To edit a mapping:

- Click the mapping name.

To delete a mapping:

- Click Delete under Actions for the mapping and then click Save.

Configuring Attribute Lookup for Token Exchange Mapping

Attribute sources are specific data store or directory locations containing information that may be required to fulfill a token-generator contract.

This optional portion of the configuration allows you to configure one or more data stores to look up attributes and to set up search parameters.
If data-store lookup is not required, click **Next**.

If data-store lookup is required, click **Add Attribute Source** and complete the setup steps (see "Choosing a Data Store for Token Mapping" next).

**To modify an attribute source configuration:**
1. Click the attribute source Description link.
2. Click **Save** on the screen you change.

**Note:** Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click **Save** or **Done** when either of those options appears.

### Choosing a Data Store for Token Mapping

On this screen choose a data store to use for supplemental user attributes.

**To define an attribute source:**
1. Enter an **Attribute Source Id** to uniquely identify the data source for the mapping.
2. Use **Attribute Source Description** to specify an attribute source name that distinguishes this user lookup for the selected data store.

**Note:** PingFederate appends this description to the data store type in the Source list on the Token Contract Fulfillment screen.

3. Choose an **Active Data Store** and click **Next**.

A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down
menu, click **Manage Data Stores** to add it (see “Managing Data Stores” on page 117).

**Note:** Data-store lookup screens for this configuration are functionally identical to those used for an IdP connection. Refer to the table below to find applicable sections in this manual containing configuration information and procedures. Then continue to the next section, “Token Contract Fulfillment”.

For information about data-store lookup configuration screens, depending on the type of data store, use this table:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Manual Section</th>
</tr>
</thead>
</table>
| JDBC            | • “Selecting a JDBC Database Table and Columns” on page 273  
                  • “Configuring a Database Filter (WHERE Clause)” on page 275 |
| LDAP            | • “Configuring an LDAP Directory Search” on page 277  
                  • “Configuring an LDAP Filter” on page 280 |
| Custom          | • “Configuring Custom Source Filters” on page 282  
                  • “Selecting Custom Source Fields” on page 282 |

**Token Contract Fulfillment**

The next step in this configuration is to map values from the token processor into the attributes required by the token generator (the Token Generator Contract).

Map each attribute to fulfill the Token Generator Contract from one of these Sources:

- **Token**
  
  When you make this selection, the associated **Value** drop-down list is populated by the token processor.
• **Context**
  
  Values are returned from the context of the transaction at runtime.

  [Note: The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.]

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

• **LDAP/JDBC/Custom (if a data store is configured)**
  
  Values are returned from a data source. When you make this selection, the Value list is populated by the LDAP, JDBC, or Custom attributes you identified.

  [Note: PingFederate appends a description in parentheses for configured data store lookups (see “Choosing a Data Store for Token Mapping” on page 147).]

• **Expression (when enabled)**
  
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

• **Text**
  
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the token processor, using the ${attribute} syntax.

  You can also enter values from your data store, when applicable, using this syntax:

  $$\text{${ds.attr-source-id.attribute}}$$

  where **attr-source-id** is the Attribute Source Id value (see “Choosing a Data Store for Token Mapping” on page 147) and **attribute** is any of the data store attributes you select.

**To map attributes:**

1. Choose a Source for each Token Generator Contract attribute.
   
   See the list under “Map each attribute to fulfill the Token Generator Contract from one of these Sources:” above.

2. Choose (or enter) a Value for each attribute.
   
   All values must be mapped.

3. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**Selecting Token Issuance Criteria**

Use this optional configuration to define criteria PingFederate can evaluate to determine whether to issue a security token (see “About Token Authorization” on
This token authorization can be used to restrict who can access protected Web services.

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   
   Associated attributes appear in the Attribute Name drop-down list:
   
   - Context – Select to use values returned from the context of the transaction at runtime.
   
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   
   - Mapped Attributes – Select to access the required attributes from the token contract.
   
   - Token – Select to access attributes from the IdP token processor.

2. Select an attribute name.

3. Select the Condition you want to apply.

   Note: Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.
4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

   Errors result in a SOAP message returned. The Error Result field is used by the faultstring element for SOAP 1.1 and the Reason/Text element for SOAP 1.2.


   If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click Add.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

   Note: All criteria must pass in order for a user to be authorized.

8. (Optional) Click Show Advanced Criteria to configure more complex evaluations using OGNL expressions.

   Note: Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

   - Use the in-line editor box to enter the OGNL expression.
     For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
   - Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

   Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

   - Click Add. Continue to add expressions, as needed.
Chapter 3  
System Settings

- Click the **Test** link to input values and test the resulting output for the expression.

  **Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

   **To edit Issuance Criteria:**
   ▶ Click **Edit** under **Actions** for the criteria.

   **To delete Issuance Criteria:**
   ▶ Click **Delete** under **Actions** for the criteria and then click **Save**.

Using the Token Mapping Summary Screen

When you have finished configuring token-to-token mapping, you can review the configuration on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Done** and then **Save** on the Manage Mappings screen.
OAuth Configuration

To use the OAuth Authorization Server (AS), start by enabling that role for PingFederate under Server Settings on the Roles and Protocols screen. Then configure the OAuth AS using options available under Server Configuration on the Main Menu, as described in this section.

For more information about the OAuth AS, see “About OAuth” on page 10.

**Tip:** Service providers may also add OAuth capabilities to the Browser SSO configuration for IdP connection partners—see “Configuring OAuth Attribute Mapping” on page 388.

Enabling the OAuth AS

You can enable the OAuth AS when you first install PingFederate (see “Running PingFederate for the First Time” in the “Installation” chapter of Getting Started). If you have already installed PingFederate or are upgrading to a new version, use the following procedure.

**Note:** OAuth AS capabilities are available under special licensing. If your license does not include the OAuth AS, please contact sales@pingidentity.com.

To enable the OAuth Authorization Server:
1. On the Main Menu under System Settings, click Server Settings.
2. Click Roles and Protocols under the Server Settings tab.
4. (Optional) Click OpenID Connect to enable that feature (see “OpenID Connect” on page 14).
5. Click Save.

Using OAuth Menu Selections

PingFederate provides a choice of ways for administrators and software developers to take advantage of OAuth AS capabilities, depending on configuration and application needs (see “About OAuth” on page 10). The Main Menu provides links to most of the configuration pathways, including global settings for the AS, client management, and required and optional mappings (see “Mapping OAuth Attributes” on page 12). (Additional optional mapping scenarios are provided for SPs as part of the IdP Connection task flow—see “Configuring OAuth Attribute Mapping” on page 388 and “Configuring an OAuth SAML Grant IdP Connection” on page 193.)

At a minimum from the Main Menu, an administrator must:
2. Configure settings for OAuth clients (see “Client Management” on page 177).
3. Configure how access tokens are handled, including setting up an Attribute Contract (see “Access Token Management” on page 165).
4. Configure a default attribute mapping from persistent grants to access tokens (see “Access Token Mapping” on page 189).

**Note:** This option appears on the Main Menu only after Access Token Management is configured.

Additionally, optional mappings may be configured, depending on applicable authentication-context mappings (see “Mapping OAuth Attributes” on page 12), as described in the following procedure.

**To configure additional, optional mappings:**

- Configure policies to extend OAuth process to include authorization policies for clients using OpenID Connect, including attribute mappings specific to that standard (see “OpenID Connect” on page 14).

  **Note:** This option appears only after the OpenID Connect protocol is enabled in Server Settings (see “Choosing Roles and Protocols” on page 106).

- Map persistent grants based on contexts, as described below, and then return to the Access Token Mapping screen to complete the mapping(s) (see “Access Token Mapping” on page 189).
  
  - “Resource-Owner Credentials Mapping” on page 181
    
    Use this option to map persistent grants based on authentication handled by PingFederate via Password Credential Validators (see “Validating Password Credentials” on page 217).
  
  - “IdP Adapter Mapping for OAuth” on page 185
    
    Use this option to map persistent grants based on user authentication information provided by PingFederate IdP adapters (see “Configuring IdP Adapters” on page 230).
  
  - “Configuring OAuth Attribute Mapping” on page 388
    
    This mapping option can be configured for Service Providers as part of an IdP connection, using incoming assertion attributes to populate persistent grants.

**Centralized Session Management**

When an organization opens Web-based protected resources to their remote employees, business partners and customers, it has limited control over the end-user devices. To minimize security risk, both the IT administrators and end users desire session management with tight controls.

PingFederate provides an asynchronous front-channel logout endpoint and a back-channel revocation Web service to help OAuth Clients using the OpenID Connect protocol, such as PingAccess, to terminate sessions when end users log out and to prevent unauthorized access until the end users log in again.

PingAccess works out-of-the-box with PingFederate, taking full advantages of these two features. For more information, see the user guide for PingAccess (documentation.pingidentity.com/display/PA/).

**Asynchronous Front-Channel Logout**

Asynchronous Front-Channel Logout provides OpenID Connect Clients the capability to initiate single logout requests to sign off associated SLO-enabled
sessions; the logout request endpoint is /idp/startSLO.ping (see “IdP Endpoints” on page 529). Optionally, clients can add end-user sessions to a revocation list on logout and query the revocation list through the Session Revocation API. For more information on how to query session status, see “Back-Channel Session Revocation” on page 158.

Tip: Information about the Asynchronous Front-Channel Logout endpoint is also available in the OpenID Connect metadata (see “OpenID Connect Metadata Endpoint” on page 551). Look for ping_end_session_endpoint in the metadata.

On a per-client basis, PingFederate can be configured to send (via the browser) logout requests to PingAccess and additional requests to other relying parties.

Note: When the PingAccess option is selected, PingFederate sends logout requests (via the browser) to an OpenID Connect logout endpoint (/pa/oidc/logout.png) in PingAccess to sign off other domains previously called by the session. For more information, see OpenID Connect Endpoints in the user guide for PingAccess.

In addition, when signing off an SLO-enabled SAML 2.0 or WS-Federation session, as the SLO request reaches the PingFederate IdP server, the same logout process applies as well. Depending on the enterprise architecture, this could further improve single sign-on and logout use cases.

To configure Asynchronous Front-Channel Logout:

1. In the Authorization Server Settings screen, select the Track User Sessions for Logout checkbox to enable Asynchronous Front-Channel Logout for OpenID Connect Clients (see “Authorization Server Settings” on page 159).
2. (Optional) Select the Revoke User Session on Logout checkbox in the Authorization Server Settings screen to add the PingFederate session to the revocation list.
3. (Optional) For each applicable client, select the PingAccess Logout Capable checkbox for PingAccess clients and enter additional endpoints at the relying parties in LOGOUT URIS for other client types as needed (see “Configuring a Client” on page 178).

Back-Channel Session Revocation

Back-Channel Session Revocation allows OpenID Connect Clients, such as PingAccess, to query the revocation status of their sessions by sending HTTP GET requests to a session revocation endpoint in PingFederate. To access the session revocation endpoint, a client must be granted access to the Session Revocation API in the PingFederate administrative console. It must also authenticate with its Client Secret or Client Certificate and include in the request the session identifier, which can be obtained from the ID Token.

Back-Channel Session Revocation also allows the clients to revoke sessions by sending HTTP POST requests to the same session revocation endpoint. This gives application developers the flexibility to revoke sessions based on the logic of their applications.

When a browser comes back to PingFederate with a revoked session, if the HTML Form Adapter is used,PingFederate assigns a new session identifier to the request and redirects the browser to the login form for the end user to sign on again. Other IdP adapters manage their own sessions; the end users may, or may not, need to re-authenticate.
For each session added to the revocation list, PingFederate retains its revocation status for a configurable lifetime. Access control and authentication requirements to revoke sessions are identical to those to query for the revocation status.

For detailed information about querying the revocation status or adding sessions to the revocation list, see “Session Revocation API” on page 568.

To configure Back-Channel Session Revocation:
1. For each applicable client, select the Grant Access to Session Revocation API checkbox (see “Configuring a Client” on page 178).
2. Select the Include Session Identifier in ID Token checkbox in OpenID Connect policies that are applied to clients supporting session revocation (see “Configuring OpenID Connect Policies” on page 170).
3. In the Authorization Server Settings screen, modify the Session Revocation Lifetime value as needed (see “Authorization Server Settings” on page 159).

**Important:** The Session Revocation Lifetime value should match or exceed the maximum session lifetime of the relying parties, including the Session Timeout value in all the applicable instances of the HTML Form Adapter (see “Configuring the HTML Form IdP Adapter” on page 515).

For example, the default Session Revocation Lifetime value of 485 minutes exceeds the default lifetime of the PingAccess (PA) Tokens by 5 minutes to allow for clock skew between servers (see Create a Web Session in the user guide for PingAccess).

### Authorization Server Settings

The Authorization Server Settings screen provides overall controls over the usage and behavior of the PingFederate OAuth AS, including scope descriptions, authorization-code policy, and refresh-token and persistent-grant policy.
### OAuth Configuration

#### Authorization Server Settings

**Scope Settings**

<table>
<thead>
<tr>
<th>Scope Value</th>
<th>Scope Description</th>
<th>Action</th>
</tr>
</thead>
</table>

**Authorization Code Policy Settings**

- **Authorization Code Timeout (seconds)**: 60
- **Authorization Code Entropy (bytes)**: 30

**Refresh Token and Persistent Grant Settings**

- **Persistent Grant Lifetime (blank for indefinite)**
- **Refresh Token Length (characters)**: 42
- **Roll Refresh Token Values (default policy)**: On
- **Minimum Interval to Roll Refresh Tokens (hours)**: 0
- **Reuse Existed Persistent Access Grants for Grant Types**: Implicit

**Bypass Authorization for Previously Approved Persistent Grants**: Off

**Allow Unidentified Clients to Make Resource Owner Password Credentials Grants**: Off

**Allow Unidentified Clients to Request Extension Grants**: Off

#### OAuth Administrative Web Services Settings

- **Password Credential Validator**: 

#### OpenID Connect Settings

- **Track User Sessions for Logout**: Off
- **Revoke User Session on Logout**: Off
- **Session Revocation Lifetime (minutes)**: 485
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Scope Description</td>
<td>Description of the permissions implied when no scope values are indicated or in addition to any values. This description displays when the user is prompted for authorization.</td>
</tr>
<tr>
<td>Tip:</td>
<td>If you want to localize the display text, you can enter a unique alias here, then use the same alias in language resource files (see “Localization” on page 92).</td>
</tr>
<tr>
<td>Scope Value</td>
<td>A value that represents access to a resource or API on the Resource Server (RS). Applicable values require coordination with a developer or someone familiar with details of the RS OAuth implementation.</td>
</tr>
<tr>
<td>Tip:</td>
<td>For OpenID Connect, you can change the message presented to users for their authorization by entering openid as a Scope Value and the message you want under Scope Description.</td>
</tr>
<tr>
<td>Important:</td>
<td>Also for OpenID Connect, to use the following key attributes in authorization policies, you must add them in this row, along with Scope Descriptions (see “Configuring OpenID Connect Policies” on page 170).</td>
</tr>
<tr>
<td></td>
<td>- profile</td>
</tr>
<tr>
<td></td>
<td>- email</td>
</tr>
<tr>
<td></td>
<td>- address</td>
</tr>
<tr>
<td></td>
<td>- phone</td>
</tr>
<tr>
<td>Note:</td>
<td>By default, scopes are available to all OAuth clients unless access is restricted on a per-client basis (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>Scope Description</td>
<td>A description of the Scope Value. This description appears when the user is prompted for authorization.</td>
</tr>
<tr>
<td>Tip:</td>
<td>If you want to localize the display text, you can enter unique aliases here, then use the same aliases in language resource files (see “Localization” on page 92).</td>
</tr>
<tr>
<td>Scope Group Value</td>
<td>A value that represents a group of scopes (a “super scope”) that you can reference in applicable OAuth 2.0 protocol interactions. The scopes within the scope group can be downgraded (upon refresh) into application-specific scopes with fewer permissions</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Scope Group Description</td>
<td>A description of the scope group value. This description appears when the user is prompted for authorization.</td>
</tr>
<tr>
<td><strong>Tip:</strong></td>
<td>If you want to localize the display text, you can enter unique aliases here, then use the same aliases in language resource files (see &quot;Localization&quot; on page 92).</td>
</tr>
<tr>
<td>Authorization Code Timeout (seconds)</td>
<td>The amount of time (in seconds) that an authorization code is considered valid.</td>
</tr>
<tr>
<td>Authorization Code Entropy (bytes)</td>
<td>The length (in bytes) of the authorization code returned to the OAuth Client.</td>
</tr>
<tr>
<td>Persistent Grant Lifetime</td>
<td>(Integer) Indicates the number of days or hours the OAuth AS stores persistent grants. Leave blank to maintain grants indefinitely.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>You can also set persistent grant expiration for specific OAuth clients, overriding this global configuration (see “Client Management” on page 177).</td>
</tr>
<tr>
<td>Refresh Token Length (characters)</td>
<td>The number of characters the OAuth AS uses to generate the refresh token returned to the client.</td>
</tr>
<tr>
<td>Roll Refresh Token Values (default policy)</td>
<td>When selected, the OAuth AS generates a new refresh token value when a new access token is obtained.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>New refresh token values are not issued during the defined interval (see the Minimum Interval to Roll Refresh Tokens box below).</td>
</tr>
<tr>
<td></td>
<td>Not selecting the checkbox means the refresh token value is used until it becomes invalid (either by manually revoking or by some other security setting that renders it invalid).</td>
</tr>
<tr>
<td>Minimum Interval to Roll Refresh Tokens (hours)</td>
<td>The minimum number of hours that must pass before a new refresh token value can be issued. Provides a way to allow for rolling a refresh token value without having to send a new value on every request.</td>
</tr>
</tbody>
</table>
## Using OAuth Menu Selections

**Reuse Existing Persistent Access Grants for Grant Types (checkboxes)**

If a client makes multiple requests for the same user and same (or lesser) scope, select the grant types you want the OAuth AS to reuse rather than creating a new grant for each request.

Reusing an existing persistent grant imposes a limit of one grant per client. For example, in the case of refresh tokens, only one (the most recently issued) is valid, and the previously issued refresh token is invalidated. If multiple instances of the same client are used regularly by the same user, the associated check box should be cleared.

When Implicit is selected, consent from the user is requested only for the first OAuth resource request associated with the grant. When Authorization Code is selected, the same is true if the checkbox Bypass Authorization for Previously Approved Persistent Grants is also selected.

### Tip:
On a per-client basis, user consent may be bypassed completely, overriding this setting (see “Configuring a Client” on page 178).

**Bypass Authorization for Previously Approved Persistent Grants (checkbox)**

When selected, consent from the user is requested only once (see “OAuth User-Facing Pages” on page 92). The user is not asked for authorization on subsequent requests until the access grant is revoked. This function applies only when using the Authorization Code grant type and when the Reuse Existing Persistent Access Grants for Grant Types checkbox is selected.

**Allow Unidentified Clients to Request Resource Owner Password Credentials Grants (checkbox)**

When selected, allows Resource Owners to obtain access tokens without defining a client.

**Allow Unidentified Clients to Request Extension Grants (checkbox)**

When selected, allows user-initiated or client-initiated events (for example, a mobile application or scheduled task) to obtain access tokens without the client presenting a `client_id` or `client_secret` for extension grant types (see “Extension Grant Types” on page 12).

**Password Credential Validator**

Required for HTTP Basic authentication if the OAuth REST Web Service is used for managing client applications (see “OAuth Client Management Service” on page 560). The service cannot be used if a validator is not provided.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reuse Existing Persistent Access Grants for Grant Types (checkboxes)</td>
<td>If a client makes multiple requests for the same user and same (or lesser) scope, select the grant types you want the OAuth AS to reuse rather than creating a new grant for each request. Reusing an existing persistent grant imposes a limit of one grant per client. For example, in the case of refresh tokens, only one (the most recently issued) is valid, and the previously issued refresh token is invalidated. If multiple instances of the same client are used regularly by the same user, the associated check box should be cleared. When Implicit is selected, consent from the user is requested only for the first OAuth resource request associated with the grant. When Authorization Code is selected, the same is true if the checkbox Bypass Authorization for Previously Approved Persistent Grants is also selected. <strong>Tip:</strong> On a per-client basis, user consent may be bypassed completely, overriding this setting (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>Bypass Authorization for Previously Approved Persistent Grants (checkbox)</td>
<td>When selected, consent from the user is requested only once (see “OAuth User-Facing Pages” on page 92). The user is not asked for authorization on subsequent requests until the access grant is revoked. This function applies only when using the Authorization Code grant type and when the Reuse Existing Persistent Access Grants for Grant Types checkbox is selected.</td>
</tr>
<tr>
<td>Allow Unidentified Clients to Request Resource Owner Password Credentials Grants (checkbox)</td>
<td>When selected, allows Resource Owners to obtain access tokens without defining a client.</td>
</tr>
<tr>
<td>Allow Unidentified Clients to Request Extension Grants (checkbox)</td>
<td>When selected, allows user-initiated or client-initiated events (for example, a mobile application or scheduled task) to obtain access tokens without the client presenting a <code>client_id</code> or <code>client_secret</code> for extension grant types (see “Extension Grant Types” on page 12).</td>
</tr>
<tr>
<td>Password Credential Validator</td>
<td>Required for HTTP Basic authentication if the OAuth REST Web Service is used for managing client applications (see “OAuth Client Management Service” on page 560). The service cannot be used if a validator is not provided.</td>
</tr>
</tbody>
</table>
Chapter 4
OAuth Configuration

To define Authorization Server settings:

- Fill in information or change defaults, as needed, and click **Save**.

Default Scope Description, Authorization Code Timeout, and Authorization Code Entropy are required (see **Field Descriptions** above).

To add a Scope:

1. Enter a value that represents access to a resource or API on the RS.

2. Enter a description of the scope. This description appears when the user is prompted for authorization.

3. Click **Add**.

To modify a Scope:

1. Click **Edit** under Action for the Scope.

2. Edit the Scope Value or Description and click **Update**.

**Caution:** Changing a Scope Value in production will cause runtime errors if a client request specifies the scope using the previous value. In addition, errors will result if a requesting OAuth client is restricted to a scope whose Value is no longer listed on this screen, unless the affected client configuration(s) are also updated (see “Client Management” next).

**Note:** If you change your mind, be sure to click the **Cancel** link in the Actions column, not the **Cancel** button, which discards any other changes you might have made.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Track User Sessions for Logout</td>
<td>When selected, PingFederate links sessions using the OpenID Connect and other federation protocols for end users. When an end user logs out of one session, PingFederate sends (via the browser) logout requests to close other associated sessions (see “Asynchronous Front-Channel Logout” on page 157).</td>
</tr>
<tr>
<td>Revoke User Session on Logout</td>
<td>When selected, PingFederate revokes the corresponding sessions on logout (see “Back-Channel Session Revocation” on page 158).</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> PingFederate always adds the session to the revocation list, even if a logout error occurs.</td>
</tr>
<tr>
<td>Session Revocation Lifetime (minutes)</td>
<td>The amount of time (in minutes) until the revoked sessions are removed from the revocation list for optimal performance.</td>
</tr>
<tr>
<td></td>
<td><strong>Important:</strong> The value should match or exceed the maximum session lifetime of the relying parties (see “Back-Channel Session Revocation” on page 158).</td>
</tr>
</tbody>
</table>
3. Click Save.

**To delete a Scope Value:**

1. Click Delete for the Scope Value.

⚠️ **Caution:** Deleting a scope in production will cause runtime errors if a client request specifies the scope. In addition, errors will result if a requesting OAuth client is restricted to a scope whose Value is no longer listed on this screen, unless the affected client configuration(s) are also updated (see “Client Management” on page 177).

2. Click Save on the Authorization Server Settings screen.

**Access Token Management**

Use this screen flow to specify how the PingFederate AS manages OAuth access tokens.

This configuration varies depending on the plug-ins deployed on your server. For information about adding a customized plug-in, please contact a support representative via the Support Center (pingidentity.com/support).

The following sections provide information for token plug-ins bundled with PingFederate:

- “Configuring Reference-Token Management” (next)
- “Configuring JSON-Token Management” on page 167

To continue, select the token-management type, if not already selected, from the drop-down list and click **Next**.

**Configuring Reference-Token Management**

On this screen, you can make changes to preset default settings for Reference Tokens (see “Token Models” on page 10) as well as set an optional maximum token lifetime.
Chapter 4
OAuth Configuration

Make changes as needed (see Field Descriptions below), or click Next to continue.

<table>
<thead>
<tr>
<th>Field Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
<td></td>
</tr>
<tr>
<td>Token Length</td>
<td>The number of characters the AS uses to define the token reference. Increasing the length will enhance token security, if desired. (Maximum: 266)</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>The amount of time (in minutes) that an access token is considered valid.</td>
</tr>
<tr>
<td>Lifetime Extension Policy</td>
<td>Indicates whether the OAuth AS should reset token lifetimes each time a token is validated. The token plug-in checks the policy before updating the lifetime of an access token. Options are: no extension policy, reset token lifetimes only for transient tokens (not backed by a persistent policy), or reset lifetimes for all tokens.</td>
</tr>
<tr>
<td>Maximum Token Lifetime</td>
<td>(Optional) Defines an absolute maximum token lifetime (in minutes) for use with the Lifetime Extension Policy. An access token's lifetime can be extended but not beyond this setting. You must specify a value that is greater than or equal to the value specified in the Token Lifetime.</td>
</tr>
</tbody>
</table>
Configuring JSON-Token Management

On this screen, you can configure settings for bearer JSON Web Tokens (JWTs) — secure, self-contained tokens that can be validated locally by the target RS (see “Token Models” on page 10).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lifetime Extension Threshold Percentage</td>
<td>When PingFederate is deployed in a cluster and token-lifetime extension is enabled, there must be a cluster-group remote procedure call (RPC) to extend the life of a token. This setting limits RPC overhead by suspending the calls until the set threshold is crossed. For example, if the token lifetime is one hour and the threshold is 50%, the lifetime will not be extended until the remaining time is less than 30 minutes. This option could potentially reduce RPC traffic between nodes by orders of magnitude while still supporting the LifeTime Extension Policy.</td>
</tr>
</tbody>
</table>

**Advanced Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode for Synchronous RPC</td>
<td>Some RPC events require that the caller get some data from the remote nodes, so the call is synchronous and blocks waiting on the responses. This configuration setting indicates whether the caller should wait for a response from all nodes in the cluster or just a majority of nodes. This is designed to eliminate the need for a complete state synchronization at startup. Synchronous RPC calls occur when a node receives a verification request for a token it does not recognize and for token issuance.</td>
</tr>
<tr>
<td>RPC Timeout</td>
<td>Timeout between cluster nodes during synchronous communication. Recommended setting is from 100 milliseconds to 1000 (1 second).</td>
</tr>
</tbody>
</table>
Chapter 4
OAuth Configuration

To configure a JWT token instance:

1. Add one or more Symmetric Keys and/or signing Certificates:

   Use the **Add a new row . . .** links, enter information, and then click **Update** under Action.

   If you are using signing certificates not yet contained in the PingFederate key store, click **Manage Signing Certificates**.

   **Note:** Signing certificate key length must be at least 2,048 bits.
2. Change or select entries for required fields and make other changes as needed (see Field Descriptions below).

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Token Lifetime (Required)</td>
<td>The amount of time, in minutes, that an access token is considered valid.</td>
</tr>
<tr>
<td>JWS Algorithm (Required)</td>
<td>The hash-based message authentication code (HMAC) or RSA signing algorithm used to protect the integrity of the token.</td>
</tr>
<tr>
<td>Active Symmetric Key ID</td>
<td>The Key ID of the key to use when producing JWTs using an HMAC-based algorithm. (Required if an HMAC JWS Algorithm is selected above.)</td>
</tr>
<tr>
<td>Include Key ID Header Parameter</td>
<td>When selected (the default), the Key ID is used in the kid header parameter for the token.</td>
</tr>
<tr>
<td>Active Signing Certificate Key ID</td>
<td>The Key ID of the key pair and certificate to use when producing JWTs using an RSA-based algorithm. (Required if an RSA JWS Algorithm is selected above.)</td>
</tr>
<tr>
<td>Include X.509 Thumbprint Header Parameter</td>
<td>When selected, the X.509 Certificate Thumbprint is used in the x5t header parameter for the token.</td>
</tr>
<tr>
<td>Client ID Claim Name</td>
<td>The name of a JWT claim used to represent the OAuth Client ID.</td>
</tr>
<tr>
<td>Scope Claim Name</td>
<td>The name of a JWT claim used to represent the scope of the grant.</td>
</tr>
</tbody>
</table>

Advanced Fields

| Space Delimit Scope Values          | When selected, indicates scope strings will be delimited by spaces rather than represented as a JSON array (the default). |
| Issuer Claim Value                  | The value of the Issuer (iss) claim in the JWT.                             |
| Audience Claim Value                | The value of the Audience (aud) claim in the JWT.                           |
| JWT ID Claim Length                 | Indicates the number of characters of the JWT ID (jti) claim in the JWT. (If 0, no claim is included.) |
| Access Grant GUID Claim Name        | The name of the JWT claim used to carry the persistent access grant GUID. If the claim is present during validation, the grant database is consulted to ensure the grant is still in force. |
| Publish Key ID X.509 URL            | Indicates whether certificates will be made accessible by Key ID at https://<pf_host>:<port>/ext/oauth/x509/kid?v=<id> |
### Defining the Access Token Attribute Contract

On this screen, create a list of attributes to be referenced in an OAuth access token.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publish Thumbprint X.509 URL</td>
<td>Indicates whether certificates will be made accessible by thumbprint at https://&lt;pf_host&gt;:&lt;port&gt;/ext/oauth/x509/x5t?v=&lt;base64url encoded SHA-1 thumbprint&gt;</td>
</tr>
</tbody>
</table>

**To add an attribute:**

1. Enter the attribute name in the text box.  
   Attribute names are case-sensitive and must correspond to the attribute names expected by the Resource Server.
2. Click **Add**.

**To modify an attribute name:**

1. Click **Edit** under Action for the attribute.
2. Make the change and click **Update**.

**To delete an attribute:**

- Click **Delete** under Action for the attribute.

### Using the Access Token Management Summary Screen

When you have finished the configuration, you can review it on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Save**.

### Configuring OpenID Connect Policies

This configuration allows you to define OpenID Connect policies for client access to attributes mapped according to OpenID specifications (see “OpenID Connect” on page 14). It also provides an option to include a session identifier in the ID Tokens, which could be useful for the relying parties, such as PingAccess (see “Back-Channel Session Revocation” on page 158).

**Note:** The OpenID Connect Policy Management link on the Main Menu appears only after the protocol is enabled in Server Settings (see “Choosing Roles and Protocols” on page 106).
To begin creating a new policy:

1. Click **Add Policy**.
2. On the next screen, **Manage Policy**, enter a **Policy ID** and **Name**.
   - The Policy ID is used internally and may not contain any spaces or underscores.
   - The Name is any you choose for display in other UI screens.
3. (Optional) Change the default **ID Token Lifetime**.
   - Values can be between 0 and 65,535
4. (Optional) Select the **Include Session Identifier in ID Token** checkbox to add a session identifier in the ID Tokens.

To edit an existing policy:

- Click its link and then on the **Summary** screen click the heading over the information that needs updating.

**Configuring the Policy Attribute Contract**

On this screen, specify attributes you want returned from the PingFederate UserInfo endpoint for OpenID Connect.

---

**Note:** The `sub` value of this configuration is also used to fulfill an internally managed ID Token attribute contract. The contract usage is dynamic, depending on how and when the user authenticated.
Chapter 4
OAuth Configuration

To add an attribute:
► Enter the attribute name in the text box and click Add.

To modify an attribute name:
1. Click Edit under Action for the attribute.
2. Make the change and click Update.

To delete an attribute:
► Click Delete under Action for the attribute.

Policy Attribute Sources and User Lookup

Attribute sources are specific data store or directory locations containing information that may be needed to fulfill the attribute contract for a policy. This configuration is optional.

► If you are not using a data store, click Next and see the next section, “Policy Contract Fulfillment”.
To configure an attribute source:

1. Click **Add Attribute Source**.
2. On the Data Store screen, choose an Active Data Store and continue the configuration.

**Note:** This setup is identical for all OAuth mapping configurations—for information, see “OAuth Attribute Mapping Using a Data Store” on page 199. When complete, click **Next** on this screen and see the next section.

**Policy Contract Fulfillment**

On this screen map attributes from the access token or other sources to fulfill the attribute contract.

<table>
<thead>
<tr>
<th>ATTRIBUTE CONTRACT</th>
<th>SOURCE</th>
<th>VALUE</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>address.locality</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>address.postal_code</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>address.street_address</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>email</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>family_name</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>given_name</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>name</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>nickname</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
<tr>
<td>phone_number</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
</tbody>
</table>
Map the subject attribute and all extended attributes from one of these Sources:

- **Context**
  Values are returned from the context of the transaction at runtime.

  ![Note:](image)
  The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click Edit to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **LDAP/JDBC/Custom (when a data store is used)**
  Values are returned from your data store (if used). When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes identified for this data store (see “Searching LDAP for OAuth Mapping” on page 201, “Defining a JDBC Location for OAuth” on page 199, or “Configuring OAuth Custom Source Filters” on page 202).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to the unique user ID returned from the credentials validator, using the syntax `${attribute}`.

  You can also enter values from your data store, when applicable, using this syntax:

  ```
  ${ds.attribute}
  ```

  where attribute is any of the data store attributes you have selected.

- **Access Token**
  The value is provided from the access token (“Access Token Management” on page 165).

To map attributes:
1. Choose a Source for the attribute.
2. Choose (or enter) a Value.
3. Click Next.

**Identifying Issuance Criteria for Policy Mapping**

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue the ID token and access token. If a criterion fails, no tokens are issued and access is denied (see “About Token Authorization” on page 24).
To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   - Context – Select to use values returned from the context of the transaction at runtime.
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   - Mapped Attributes – Select to access the output data returned from UserInfo endpoint.
   - Access Token – Select to use attributes from the access token.

2. Select an attribute name.

3. Select the Condition you want to apply.

4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.
   Errors result in the value of this field being used by the error_description protocol field.

   **Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

   If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click Add.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

- **Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations (for example, OR conditions) using OGNL expressions.

- **Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

   - Use the in-line editor box to enter the OGNL expression.
   - For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
   - Use the Error Result box to enter an error message or an error code for use if authorization fails (see step 5 above for more information).

- **Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

   - Click **Add**. Continue to add expressions, as needed.
   - Click the **Test** link to input values and test the resulting output for the expression.

- **Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Policies screen).

**To edit Issuance Criteria:**
- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**
- Click **Delete** under Actions for the criteria and then click **Save**.

**Using the Policy Summary Screen**

When you have finished the configuration, you can review it on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Done** and then **Save** on the Manage Policies screen.

- **Note:** If this is the first policy you are creating, you must designate it as the default before saving. You can change the default as needed when you create additional policies.
Client Management

An OAuth client application interacts with the PingFederate OAuth AS to obtain access tokens needed to call OAuth-protected services at the Resource Server. Use this screen flow to specify how the OAuth AS manages these applications.

**Important:** OAuth client records, like other PingFederate configuration data, are maintained in XML files. Alternatively, you can set up a centralized database to store client records (see “Defining an OAuth Client Data Store” on page 131).

**Note:** When a database is defined, PingFederate provides a REST Web Service for managing OAuth clients, allowing administrators to add and maintain clients in a database programmatically (see “OAuth Client Management Service” on page 560).

**Tip:** For detailed developer information for client applications, see “OAuth 2.0 Endpoints” on page 543.

To add a new Client:

- Click **Add Client**.

To edit an existing Client:

1. Click the Client ID and make the necessary changes on the Manage Clients screen.

**Tip:** Use the **Search** features to locate clients that may be out of view or on subsequent pages, or use the page-navigation controls, when present. Click **Clear** to redisplay the list after a search. (Select Advanced Search for search options.)

2. Click **Done** and then **Save**.

To delete a Client:

1. Click **Delete** next to the Client name. (To undo the deletion, click **Undelete**.)
2. Click **Save** to confirm the deletion.
Configuring a Client

The Manage Client screen provides controls over the usage and behavior of the applications requesting access to protected resources through the PingFederate AS.

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client ID</td>
<td>(Required) A unique identifier the client provides to the Resource Server to identify itself. This identifier is included with every request the client makes.</td>
</tr>
</tbody>
</table>
### Using OAuth Menu Selections

#### Client Authentication

A selection other than **None** is required only for the Client Credentials grant type (see “Grant Types” on page 11).

When required, select *either*:

- **Client Secret** for Basic authentication.
  
  Click **Generate Secret** to create a strong random alphanumeric string or manually enter a secret. If this button is disabled, select the **Change Secret** checkbox.

  *Or:*

- **Client Certificate** for mutual SSL/TLS authentication—recommended for application configurations where security policies prohibit storing passwords.
  
  **Note:** Mutual SSL/TLS authentication requires the use of a secondary PingFederate SSL port (see “Changing Configuration Parameters” on page 74).

Enter information into the following required fields:

- **Client TLS Certificate Issuer** – Select a Trusted CA from the list (see “Trusted Certificate Authorities” on page 204). Alternatively, you may choose to Trust Any of the issuers listed (CA certificates imported into PingFederate).

- **Client TLS Certificate Subject DN** – Enter the client-certificate subject DN, or click **Browse** to locate the certificate and then **Extract** to retrieve the DN.

#### Name

(Required) A descriptive name for the client instance. This name appears when the user is prompted for authorization.

**Tip:** If you want to localize the displayed name, you can enter a unique alias here, then use the same alias in language resource files (see “Localization” on page 92).

#### Description

A description of what the client application does. This description appears when the user is prompted for authorization.

**Tip:** If you want to localize the displayed description, you can enter a unique alias here, then use the same alias in language resource files (see “Localization” on page 92).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>(Required) A descriptive name for the client instance. This name appears when the user is prompted for authorization. <strong>Tip:</strong> If you want to localize the displayed name, you can enter a unique alias here, then use the same alias in language resource files (see “Localization” on page 92).</td>
</tr>
<tr>
<td>Description</td>
<td>A description of what the client application does. This description appears when the user is prompted for authorization. <strong>Tip:</strong> If you want to localize the displayed description, you can enter a unique alias here, then use the same alias in language resource files (see “Localization” on page 92).</td>
</tr>
</tbody>
</table>
Chapter 4
OAuth Configuration

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redirection URIs</td>
<td>URIs to which the OAuth AS may redirect the resource owner’s user agent after authorization is obtained. A redirection URI is used with the Authorization Code and Implicit grant types (see “Grant Types” on page 11). Enter a fully qualified URL and click Add for each entry required. Wildcards are allowed. However, for security reasons, make the URL as restrictive as possible. For example: https://<em>.company.com/</em> <strong>Important</strong>: If more than one URI is added or if a single URI uses wildcards, then Authorization Code grant and token requests must contain a specific matching redirect_uri parameter (see “OAuth 2.0 Endpoints” on page 543).</td>
</tr>
<tr>
<td>Logo URL</td>
<td>The location of the logo used on user-facing OAuth grant authorization and revocation pages (see “Customizing User-Facing Screens” on page 87). (For best results with the installed HTML templates, the recommended size is 72 x 72 pixels.)</td>
</tr>
<tr>
<td>Bypass Authorization Approval</td>
<td>When selected, resource-owner approval for client access is assumed—the authorization consent page is not presented to the user for this client. Use this setting, for example, when you want to deploy a trusted application and authenticate end users via an IdP adapter or IdP connection.</td>
</tr>
<tr>
<td>Restrict Scopes</td>
<td>Restricts this client’s access to specific scopes or scope groups. When selected, a list appears of all scopes defined within the PingFederate AS. Select the scopes or scope groups available for this client. <strong>Note</strong>: Selecting this box and not selecting any specific scopes restricts this client’s access to only the default scope (see “Authorization Server Settings” on page 159). If a client requests a specific scope (not selected in the list), an error is returned.</td>
</tr>
<tr>
<td>Allowed Grant Types</td>
<td>Available grant types that this client is allowed to use.</td>
</tr>
<tr>
<td>Persistent Grants Expiration</td>
<td>Allows an administrator to override the Persistent Grant Lifetime set globally for the OAuth AS (see “Authorization Server Settings” on page 159).</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refresh Token Rolling Policy</td>
<td>Select <strong>Don’t Roll</strong> or <strong>Roll</strong> to override the Roll Refresh Token Values setting on the Authorization Server Settings screen (see “Authorization Server Settings” on page 159). Leave <strong>Use Global Setting</strong> selected to default to the Roll Refresh Token Values setting on the Authorization Server Setting screen.</td>
</tr>
</tbody>
</table>

**OpenID Connect**

**Note:** These options are displayed only when OpenID Connect is enabled in Server Settings (see “Choosing Roles and Protocols” on page 106).

**ID Token Signing Algorithm:** (Optional) Select the signing algorithm for the ID Token. The default algorithm is RS256.

**Policy:** (Optional) Choose a specific OpenID Connect policy (see “Configuring OpenID Connect Policies” on page 170).

**Grant Access to Session Revocation API:** When selected, this client application is allowed to add sessions to or query the revocation status. Authentication is required (see “Session Revocation API” on page 568).

**Note:** If the Track User Sessions for Logout checkbox is selected in the Authorization Server Settings screen, there are two additional settings:

**PingAccess Logout Capable:** (Optional) When selected, PingFederate sends (via the browser) logout requests to an OpenID Connect endpoint in PingAccess as part of the logout process (see “Asynchronous Front-Channel Logout” on page 157).

**Logout URIs:** (Optional) Enter additional endpoints at the relying parties as needed. PingFederate sends (via the browser) requests to these URIs as part of the logout process (see “Asynchronous Front-Channel Logout” on page 157).

For more information about the logout endpoint in PingAccess, see OpenID Connect Endpoints in the user guide for PingAccess.

---

**Resource-Owner Credentials Mapping**

This configuration allows you to map attributes based on validated user credentials into the **USER_KEY** for a persistent grant (see “Mapping OAuth Attributes” on...
You may supplement credential-validation mapping sources with attribute look-ups from your user data source.

**Note:** At least one credential validator instance must be configured in order to map attributes (see “Validating Password Credentials” on page 217).

To create a token mapping:
1. Select a Source Instance from the drop-down list and click Add Mapping.
   - If the list does not contain any instances or the instance you want, return to the Main Menu and click Password Credential Validators under Authentication (see “Validating Password Credentials” on page 217).
2. (Optional) On the Data Store screen, choose an Active Data Store.

**Note:** The Data Store screen provides an option to look up alternate attributes based on the ID returned from the validator instance. The setup is identical for all OAuth mapping configurations—for information, see “OAuth Attribute Mapping Using a Data Store” on page 199.

If you are not using a data store, see the next section, “Resource-Owner Contract Fulfillment”.

To edit an existing mapping:
- Click its link and then on the Summary screen click the heading over the information that needs updating.

**Resource-Owner Contract Fulfillment**

On this screen, you map values into the USER_KEY for a persistent grant. Use this mapping for the Resource Credential grant type (see “Grant Types” on page 11).
Map an attribute to populate the USER_KEY from one of these Sources:

- **Context**
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **Password Credential Validator**
  When you make this selection, the associated Value drop-down list consists of the attributes (for example, username) associated with the credential-validation instance.

- **LDAP/JDBC/Custom (when a data store is used)**
  Values are returned from your data store (if used). When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes identified for this data store (see “Searching LDAP for OAuth Mapping” on page 201, “Defining a JDBC Location for OAuth” on page 199, or “Configuring OAuth Custom Source Filters” on page 202).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to the unique user ID returned from the credentials validator, using the syntax `${attribute}`.

  You can also enter values from your data store, when applicable, using this syntax:

  ```
  ${ds.attribute}
  ```

  where `attribute` is any of the data store attributes you have selected.

**To map attributes:**

1. Choose a **Source** for the attribute.
   (For descriptions, see “Map an attribute to populate the USER_KEY from one of these Sources:” above.)

2. Choose (or enter) a **Value**.

3. Click **Next**.

**Identifying Issuance Criteria for Credentials Mapping**

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue an access token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access an OAuth-protected resource.
To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   
   Associated attributes appear in the Attribute Name drop-down list:
   
   • Context – Select to use values returned from the context of the transaction at runtime.
   
   • LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   
   • Mapped Attributes – Select to access the output of the mapping into the persistent grant—for example, the USER_KEY.
   
   • Password Credential Validator – Select to access attributes returned from the credential-validation instance.

2. Select an attribute name.

3. Select the Condition you want to apply.

   • Note: Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

   Errors result in the value of this field being used by the error_description protocol field.

   • Note: Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

   If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click Add.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

Note: All criteria must pass in order for a user to be authorized.

8. (Optional) Click Show Advanced Criteria to configure more complex evaluations (for example, OR conditions) using OGNL expressions.

Note: Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

- Use the in-line editor box to enter the OGNL expression. 
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the Error Result box to enter an error message or an error code for use if authorization fails (see step 5 above for more information).

Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click Add. Continue to add expressions, as needed.
- Click the Test link to input values and test the resulting output for the expression.

Note: For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click Next (or Done, if you are finished with the configuration, and then Save on the Manage Mappings screen).

To edit Issuance Criteria:
- Click Edit under Actions for the criteria.

To delete Issuance Criteria:
- Click Delete under Actions for the criteria and then click Save.

Using the Credentials-Mapping Summary Screen

When you have finished the configuration, you can review it on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click Done and then Save on the Manage Mappings screen.

IdP Adapter Mapping for OAuth

This configuration allows you to map attributes based on an IdP adapter configuration into the USER_KEY for a persistent grant and USER_NAME (presented to the user for authorization permission). You may supplement values returned from the adapter with attribute look-ups from your user data source. (For more information about adapters, see “Configuring IdP Adapters” on page 230.)
Use this mapping for Authorization Code and Implicit grant types (see “Grant Types” on page 11).

To create a token mapping:
1. Select a Source Instance from the drop-down list and click Add Mapping.
2. (Optional) On the Data Store screen, choose an Active Data Store.

**Note:** The Data Store screen provides an option to look up alternate attributes based on attributes returned from the IdP adapter instance. The setup is identical for all OAuth mapping configurations—for information, see “OAuth Attribute Mapping Using a Data Store” on page 199.

If you are not using a data store, see the next section, “Grant Contract Fulfillment”.

**To edit an existing mapping:**
> Click its link and then on the Summary screen click the heading over the information that needs updating.

**Grant Contract Fulfillment**

On this screen, you map values into the USER_KEY and USER_NAME for the persistent grant and the user’s display name on the authorization page, respectively.

Map an attribute to populate the USER_KEY from one of these Sources:
- Adapter

When you make this selection, the associated Value drop-down list contains attributes configured in the IdP adapter instance.
Using OAuth Menu Selections

- Context
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- LDAP/JDBC/Custom (when a data store is used)
  Values are returned from your data store (if used). When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes identified for this data store (see “Searching LDAP for OAuth Mapping” on page 201, “Defining a JDBC Location for OAuth” on page 199, or “Configuring OAuth Custom Source Filters” on page 202).

- Expression (when enabled)
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- Text
  The value is what you enter. This can be text only, or you can mix text with references to the attributes returned from the adapter instance, using the syntax `${attribute}`.

  You can also enter values from your data store, when applicable, using this syntax:

  
  \[ ${ds.attribute} \]

  where **attribute** is any of the data store attributes you have selected.

**To map attributes:**

1. Choose a Source for the attribute.

   (For descriptions, see “Map an attribute to populate the USER_KEY from one of these Sources:” above.)

2. Choose (or enter) a Value.

3. Click **Next**.

**Defining Issuance Criteria for OAuth Adapter Mapping**

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue an access token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access an OAuth-protected resource.
To configure Issuance Criteria:

1. Select a source containing attributes for token authorization. Associated attributes appear in the Attribute Name drop-down list:
   - Adapter – Select to access attributes from the IdP Adapter.
   - Context – Select to use values returned from the context of the transaction at runtime.
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   - Mapped Attributes – Select to access the output of the mapping into the persistent grant—for example, the USER_KEY.

   **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   - Mapped Attributes – Select to access the output of the mapping into the persistent grant—for example, the USER_KEY.

2. Select an attribute name.
3. Select the Condition you want to apply.

   **Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

   Errors result in the value of this field being used by the error_description protocol field.

   **Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

   If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click **Add**.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

- Use the in-line editor box to enter the OGNL expression.
  
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.

- Use the Error Result box to enter an error message or an error code for use if authorization fails (see step 5 for more information).

**Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click **Add**. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**

- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**

- Click **Delete** under Actions for the criteria and then click **Save**.

**Using the Adapter-Mapping Summary Screen**

When you have finished the configuration, you can review it on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Done** and then **Save** on the Manage Mappings screen.

**Access Token Mapping**

In this required configuration, an administrator maps attributes to be requested from the OAuth resource server with the access token—the token attribute contract (see “Defining the Access Token Attribute Contract” on page 170). In the configuration, you define one or more contexts for controlling how the OAuth AS maps values into the attributes based on the persistent-grant **USER_KEY**.
Chapter 4

OAuth Configuration

The mapping used at runtime depends on the authentication context of the original grant.

To create a token mapping:

1. Select a Context from the drop-down list and click Add Mapping.

   Contexts for mappings include:
   - (Required) The default mapping when no other contexts are configured
   - The user key for a grant request as determined by instances of a credentials validator (see “Validating Password Credentials” on page 217)
   - User key from the OAuth IdP adapter-mapping configurations (see “IdP Adapter Mapping for OAuth” on page 185)
   - SP-connection OAuth mappings (see “Configuring OAuth Attribute Mapping” on page 388)

2. (Optional) On the Data Store screen, choose an Active Data Store.

To edit an existing mapping:

- Click its link and then on the Summary screen click the heading over the information that needs updating.

Note: This option appears on the Main Menu only after Access Token Management is configured (see “Access Token Management” on page 165).

Token Attribute Contract Fulfillment

On this screen, you map values into the token attribute contract (see “Defining the Access Token Attribute Contract” on page 170). These are the attributes that will be included or referenced in the access token.

Note: The Data Store screen provides an option to look up additional attributes to map into the token attribute contract. Data-store setup is identical for all OAuth mapping configurations—for information, see “OAuth Attribute Mapping Using a Data Store” on page 199.

If you are not using a data store, see the next section, “Token Attribute Contract Fulfillment”.

To edit an existing mapping:

- Click its link and then on the Summary screen click the heading over the information that needs updating.
Map each attribute to fulfill the Token Attribute Contract from one of these Sources:

- **Context**
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **Persistent Grant**
  When you make this selection, the associated **Value** drop-down list is populated by the USER_KEY from the persistent access-token grant.

- **LDAP/JDBC/Custom (when a data store is used)**
  Values are returned from your user-data store. When you make this selection, the **Value** list is populated by the LDAP, JDBC or Custom attributes identified for this data store (see “Searching LDAP for OAuth Mapping” on page 201, “Defining a JDBC Location for OAuth” on page 199, or “Configuring OAuth Custom Source Filters” on page 202).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to the USER_KEY using the syntax ${USER_KEY}.

  You can also enter values from your data store, when applicable, using this syntax:

  ```
  ${ds.attribute}
  ```

  where **attribute** is any of the data store attributes you have selected.

**To map attributes:**

1. Choose a **Source** for each attribute.
   (For descriptions, see “Map each attribute to fulfill the Token Attribute Contract from one of these Sources:” above.)

2. Choose (or enter) a **Value** for each Attribute.
   All values must be mapped.

3. Click **Next**.

**Configuring Issuance Criteria for Access Token Mapping**

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue an access token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access an OAuth-protected resource. For example, before reissuing a token in a refresh-token scenario,
the server can verify that the user is still current in your organization’s user-data store and has retained the necessary permissions.

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   - Context – Select to use values returned from the context of the transaction at runtime.
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   - Mapped Attributes – Select to use access token attributes returned from token attribute contract mapping.
   - Persistent Grant – When you select this option, the associated Attribute Name drop-down list is populated by the USER_KEY from the persistent access-token grant.

2. Select an attribute name.
3. Select the Condition you want to apply.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.
   Errors result in the value of this field being used by the error_description protocol field.

**Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click Add.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

   **Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

   **Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

   - Use the in-line editor box to enter the OGNL expression.
     For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
   - Use the Error Result box to enter an error message or an error code for use if authorization fails (see step 5 for more information).

   **Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

   - Click **Add**. Continue to add expressions, as needed.
   - Click the **Test** link to input values and test the resulting output for the expression.

   **Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

   **To edit Issuance Criteria:**
   - Click **Edit** under Actions for the criteria.

   **To delete Issuance Criteria:**
   - Click **Delete** under Actions for the criteria and then click **Save**.

Using the Token-Mapping Summary Screen

When you have finished the configuration, you can review it on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Done** and then **Save** on the Manage Mappings screen.

### Configuring an OAuth SAML Grant IdP Connection

An OAuth SAML Grant connection exchanges a SAML assertion for an OAuth access token with the PingFederate Authorization Server. You can configure an OAuth SAML Grant connection with an IdP partner either in conjunction with browser-based SSO, WS-Trust, or independently.
To enable OAuth SAML Grant for a new connection, or to add the capability to an existing connection:

- On the Connection Type screen, select OAuth SAML Grant (see “Choosing an IdP Connection Type” on page 358).

Note: Before you can select this option, you must enable the OAuth 2.0 protocol in Server Settings (see “Choosing Roles and Protocols” on page 106) and configure an access token plug-in (see “Access Token Management” on page 165).

When the option is enabled, the configuration starts on the OAuth SAML Grant Attribute Mapping screen.

- To continue, click Configure OAuth SAML Grant Attribute Mapping.

### Specifying an Attribute Contract for the OAuth SAML Grant

An attribute contract is a set of user attributes the IdP sends in the SAML assertion for this connection. You identity these attributes on this screen.

SAML_SUBJECT is always sent in a SAML assertion and contains the name identifier of the user for whom the access token is being requested.

Optionally, you can mask the values of attributes (other than SAML_SUBJECT) in the log files that PingFederate writes when it receives security tokens (see “Attribute Masking” on page 23).

To add an attribute:

1. Enter the attribute name in the text box.
   Attribute names are case-sensitive and must correspond to the attribute names expected by your partner.
2. (Optional) Select the checkbox under Mask Values in Log.
3. Click Add.
Configuring an OAuth SAML Grant IdP Connection

To modify an attribute name or masking selection:
1. Click **Edit** under Action for the attribute.
2. Make the change and click **Update**.

**Note:** If you change your mind, ensure that you click the Cancel link in the Actions column, not the Cancel button, which discards any other changes you might have made in the configuration steps.

To delete an attribute:
- Click **Delete** under Action for the attribute.

Choosing a Data Store for OAuth SAML Grant Attribute Mapping

This optional configuration is the same for all OAuth attribute-mapping task flows. For detailed instructions, see “OAuth Attribute Mapping Using a Data Store” on page 199.

- If you do not need additional attributes from a data store, just click **Next** on the Data Store screen.

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **OAuth SAML Grant Attribute Mapping** under the IdP Connection tab.
3. Click **Configure OAuth SAML Grant Attribute Mapping**.
4. Click **Data Store** on the Summary screen.

OAuth SAML Grant Contract Fulfillment

The last step in configuring OAuth SAML Grant attribute mapping is to map SAML grant attributes to the attribute values of the access token (see “Access Token Mapping” on page 189).

<table>
<thead>
<tr>
<th>CONTRACT</th>
<th>SOURCE</th>
<th>VALUE</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>uid</td>
<td>Assertion</td>
<td>SAML_SUBJECT</td>
<td>None available</td>
</tr>
</tbody>
</table>
Map attributes from one of the following Sources:

- **Assertion**
  Values are contained in the assertions from this IdP. When you make this selection, the associated Value drop-down list is populated by the attribute contract.

- **Context**
  Values are returned from the context of the transaction at runtime.

**Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **JDBC/LDAP/Custom**
  Values are returned from your user-data store. When you make this selection, the Value list is populated by the LDAP, JDBC, or Custom attributes identified for this data store (see “Defining a JDBC Location for OAuth” on page 199 or “Configuring an OAuth Database Filter (WHERE Clause)” on page 201).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the assertion, using the ${attribute} syntax.
  You can also enter values from your data store, when applicable, using this syntax:
  $$\{ds.attribute\}$$
  where attribute is any of the data store attributes you have selected.

**To reach this screen:**

1. Click the connection name on the Main Menu.
   Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **OAuth SAML Grant Attribute Mapping Configuration** under the IdP Connection tab.
3. Click **Configure OAuth SAML Grant Attribute Mapping Configuration**.
4. Click **Contract Fulfillment**.

**To map attributes:**

1. Choose a Source for each attribute.
   (For descriptions, see “Map attributes from one of the following Sources” above.)
2. Choose (or enter) a Value for each attribute.
   All values must be mapped.

3. Click **Next**.

**Selecting Issuance Criteria for OAuth SAML Grant**

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue an access token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access an OAuth-protected resource.

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   - **Assertion** – Select to access attributes from the assertion.
   - **Context** – Select to use values returned from the context of the transaction at runtime.

   **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

2. Select an attribute name.

3. Select the Condition you want to apply.

   **Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

Errors result in the value of this field being used by the error_description protocol field.

**Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click **Add**.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

- Use the in-line editor box to enter the OGNL expression.
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the Error Result box to enter an error message or an error code for use if authorization fails (see step 5 above for more information).

**Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click **Add**. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**
- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**
- Click **Delete** under Actions for the criteria and then click **Save**.
OAuth SAML Grant Configuration Summary

When you finish the configuration, you can review it on the Summary screen. If you need to make any changes, click the heading over the information you want to edit. When you finish, click Done.

OAuth Attribute Mapping Using a Data Store

This optional configuration is the same for all of the OAuth attribute-mapping task flows described under “Using OAuth Menu Selections”, including:

- Configuring OpenID Connect Policies
- Resource-Owner Credentials Mapping
- IdP Adapter Mapping for OAuth
- Access Token Mapping

The same configuration is also used for attribute mapping in an IdP connection (see “Configuring OAuth Attribute Mapping” on page 388) and when configuring an OAuth SAML Grant connection (see “Configuring an OAuth SAML Grant IdP Connection” on page 193).

- If you do not need additional attributes from a data store for the respective mapping configuration, just click Next when you reach the Data Store screen.

To look up user attributes for an OAuth mapping configuration:

1. Choose an Active Data Store and click Next.

A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click Manage Data Stores to add it (see “Managing Data Stores” on page 116).

2. See the following sections, depending on the type of data store:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Related Section</th>
</tr>
</thead>
</table>
| JDBC            | “Defining a JDBC Location for OAuth” on page 199  
|                 | “Configuring an OAuth Database Filter (WHERE Clause)” on page 201 |
| LDAP            | “Searching LDAP for OAuth Mapping” on page 201  
|                 | “Configuring an LDAP Filter for OAuth Mapping” on page 202 |
| Custom          | “Configuring OAuth Custom Source Filters” on page 202  
|                 | “Selecting OAuth Custom Source Fields” on page 202 |

Defining a JDBC Location for OAuth

When you choose to use a database source for attributes, you follow this path through the configuration steps.
On the Database Tables and Columns screen you begin to specify exactly where additional data can be found to complete the attribute contract. Only one table may be used as a source of data for a JDBC lookup.

**Important: (For MySQL users)** To allow for table and column names that may contain spaces, PingFederate inserts double quotes around the names at runtime. To avoid SQL syntax errors resulting from the quotes, add the property `ANSI_QUOTES` to `sql-mode` in the configuration file `my.cnf` (on Unix/Linux) or `my.ini` (on Windows). For example:

```
sql-mode="...,ANSI_QUOTES"
```

For more information, see:

- [dev.mysql.com/doc/refman/5.0/en/identifiers.html](https://dev.mysql.com/doc/refman/5.0/en/identifiers.html)

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Lists the table structure that stores information within a database. Some databases, such as Oracle, require selection of a specific schema for a JDBC query. Other databases, such as MySQL, do not require selection of a schema.</td>
</tr>
<tr>
<td>Table</td>
<td>The name of the table contained in the database. Use the drop-down to change the table.</td>
</tr>
<tr>
<td>Columns to return from SELECT</td>
<td>Displays selected table columns. Select the columns associated with the desired attributes you would like to return from the JDBC query.</td>
</tr>
</tbody>
</table>

**To select a database table and columns for queries:**

1. Choose a Schema file (when applicable) from the drop-down list.
2. Choose a Table from the drop-down list.
3. Choose a name under Columns to Return from Select and click **Add Attribute**.

**Tip:** Click **Refresh** if you are updating an existing configuration and changes may have been made to the database.

Repeat this step for other columns as needed.

**Note:** You do not need to add a column here for it to be used as part of a search filter (see “Configuring an OAuth Database Filter (WHERE Clause)” next).
Configuring an OAuth Database Filter (WHERE Clause)

The JDBC WHERE clause in PingFederate queries the data table you selected to retrieve a record associated with a particular value (or values) from the assertion. The clause is in the form:
```
WHERE column1=value1 [AD column2=value2] [O...]
```

The left side of the first variable pair uses a column name in the database table you selected (see “Defining a JDBC Location for OAuth” on page 199).

You can also apply additional search criteria from your own database, using any other columns from the targeted table.

**Tip:** Click “View List of Columns . . .” to see a list from which to copy and paste.

For more information about WHERE clauses, consult your DBMS documentation.

To construct the WHERE clause:

1. Enter the statement in the space provided, following the guidelines and example above.
   
   **Tip:** The initial WHERE is optional.

2. Ensure the syntax and variable names are correct.

Searching LDAP for OAuth Mapping

When you choose to use an LDAP source for attributes, you follow this path through the configuration steps.

On this screen you specify the branch of the LDAP hierarchy where you want PingFederate to look up user data.

**Field Descriptions**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. This field is optional if records are located at the LDAP root.</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Determines the node depth of the query. Select Subtree, One level or Object.</td>
</tr>
<tr>
<td>Root Object Class</td>
<td>The class containing the attributes you want.</td>
</tr>
<tr>
<td>Attributes to return from search</td>
<td>A list of attributes added from the drop-down list below. Subject DN is a default attribute, which may be used as the primary user identifier.</td>
</tr>
</tbody>
</table>
To select LDAP attributes:

1. (Optional) Enter a Base DN.
2. Select a Search Scope.
3. Select a Root Object Class.
4. Under Attributes to return from search, choose an attribute and click Add Attribute.
   Note that the attribute Subject DN is always returned by default.
5. Repeat the last step for other attributes as needed.
6. (Optional) Change the Search Scope or the Root Object Class if you want attributes from other locations.

**Note:** You do not need to add an attribute here for it to be used in a search filter (see “Configuring an LDAP Filter for OAuth Mapping”). Add only attributes from which you need actual values.

Configuring an LDAP Filter for OAuth Mapping

The LDAP filter queries the data you selected to retrieve a record associated with a particular value (or values) from the user’s session. The filter is in the form:

\{attribute=${value}\}

The left-side variable is an attribute you selected earlier (see “Searching LDAP for OAuth Mapping” on page 201).

The right side generally uses values passed in from the mapping context (variables, including the correct syntax, are listed under “Values available…”)

You can also apply additional search criteria from your data store, using any other attributes from the targeted object classes.

**Tip:** Click “View List of Available LDAP Attributes” for a list from which you can copy and paste.

For general information about search filters, consult your LDAP documentation.

Configuring OAuth Custom Source Filters

When you choose to use a custom source for attributes, you follow this path through the configuration steps.

On this screen you specify a filter, or lookup query, for your custom data source. This screen display and the syntax of the filter depends on your developer’s implementation of the custom source SDK.

Selecting OAuth Custom Source Fields

On the Configure Custom Source Fields screen, you can choose from among the fields shown to map to the adapter contract. These choices are supplied by the driver implementation. Select only those needed to fulfill the token contract for this partner connection.
PingFederate provides built-in certificate management to handle SSL/TLS server security, as well as certificate signing and verification of SSO and other transactions, when required.

In addition, the server provides authentication capabilities for applications making use of secured system features, or for protocol features requiring management and validation of end-user password credentials.

This section covers:
- “Certificate Management” (next)
- “Authentication” on page 215

**Note:** This information is presented from the viewpoint of an administrative user with “Crypto Admin” permissions (see “Account Management” on page 65).

### Certificate Management

PingFederate administrators manage certificates via the Security section under Server Configuration on the Main Menu.
Chapter 5  
Security Management  

**Trusted Certificate Authorities**

You can import your federation partner's CA certificate or self-signed certificate(s) into PingFederate's global trust list. If the Certificate Authority is not one of the major authorities, you may also need to import the certificate from the CA that signed the partner certificate.

*Note:* If a required CA certificate is already available in `cacerts` in the Java runtime, it is not necessary to import the same certificate into the PingFederate store.

To import a certificate:
1. Click **Import**.
2. Click **Browse** to locate the certificate.
3. Highlight the file and click **Open**.
4. Click **Next**.
5. Click **Done**.
6. Click **Save** on the Manage Trusted CAs screen.

To export a certificate:
1. Click **Export** under Action for the certificate you want to export.
2. On the Summary page, click the **Export** button.
3. Save the file on your system.

To delete a certificate:
1. Click **Delete** under Action for the certificate you want to delete.
   To undo the deletion, click **Undelete**.
2. Click **Save**.

To view certificate details:
- Click the certificate Serial number.

**SSL Server Certificates**

PingFederate provides built-in SSL/TLS certificate management. Use this feature to establish and maintain the certificate(s) presented for access to the PingFederate
Certificate Management

administrative console and for incoming SSL/TLS connections at runtime (see “Setting Administration Options” on page 98).

To create a new certificate:
1. Click Create New.
2. Enter the requested information on the form.
3. Click Next.
4. On the Summary screen, click Done.
5. Click Save on the Manage SSL Server Certificates screen.

To import a certificate and private key:
1. Click Import.
2. Click Browse to locate the certificate.
3. Highlight the file and click Open.
4. Enter the certificate password.
5. Click Next.
6. Click Done.
7. Click Save on the Manage SSL Server Certificates screen.

To view certificate information:
- Click its Serial number.

**Note:** If a certificate has been revoked, PingFederate indicates this problem in the certificate information window.

To activate a certificate:
1. Click Activate for Runtime Server or Activate for Admin Console under Action for the certificate you want to activate.

   These choices are enabled only if you have created or imported more than one certificate. Otherwise, a single certificate is used for both the administrative console and runtime operations.

2. Click Save on the Manage SSL Server Certificates screen.
To export a certificate:
1. Click Export under Action for the certificate you want to export.
   Or:
   Select Certificate and Private Key and enter an Encryption Password.
3. Click Next.
5. Save the file on your system and click Done.

To create a certificate-authority signing request:
1. Click Certificate Signing under Action for the desired certificate.

   \[\text{Note: This selection is inactive if you have not yet saved a newly created or imported certificate. Click Save and then return to this screen from the Main Menu.}\]

   \[\text{Tip: The selection is also inactive if a previously signed certificate has been revoked. Because the revocation may indicate that the private key has been compromised, the best practice is to import or create a replacement certificate for CA signing.}\]

2. Select Generate Certificate Signing Request (CSR), if not already selected.
3. Click Next.
4. Click Generate CSR on the Generate CSR screen.
5. Click Next.
7. Save the file on your system and click Done.

To import a certificate authority response:
1. Click Certificate Signing under Action for the relevant certificate.
2. Select Import CSR Response and click Next.
3. Click Browse and locate the CSR response to import.
4. Highlight the file and click Open.
5. Click Next.
6. Click Done on the Summary screen.
7. Click Save on the Manage SSL Server Certificates screen.

To delete a certificate:
1. Click Delete under Action for the certificate you want to delete.

   \[\text{Note: This option does not appear if the certificate is in use. To enable deletion, add (if needed) and activate a different certificate for the administrative console and/or the runtime server. If the usage is not clear, click Check Usage.}\]

2. Click Save.
Create Certificate Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Name</td>
<td>The common name (CN) identifying the certificate.</td>
</tr>
<tr>
<td>Organization</td>
<td>The organization (O) or company name creating the certificate.</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>The specific unit within the organization (OU).</td>
</tr>
<tr>
<td>City</td>
<td>The city or other primary location (L) where the company operates.</td>
</tr>
<tr>
<td>State</td>
<td>The state (ST) or other political unit encompassing the location.</td>
</tr>
<tr>
<td>Country</td>
<td>The country (C) where the company is based.</td>
</tr>
<tr>
<td>Validity (days)</td>
<td>The time during which the certificate is valid.</td>
</tr>
<tr>
<td>Key Algorithm</td>
<td>A mathematical formula used to generate a key. PingFederate uses either of two algorithms, RSA or DSA.</td>
</tr>
<tr>
<td>Key Size (bits)</td>
<td>The number of bits used in the key. (RSA-768 to 2048, DSA-768 and 1024)</td>
</tr>
</tbody>
</table>

SSL Client Keys and Certificates

You can create and manage your authentication private keys and the certificates your server presents as a client in an SSL/TLS transaction.

To create a new certificate:
1. Click **Create New**.
2. Enter the information on the form.
3. Click **Next**.
4. On the Summary screen, click **Done**.
5. Click **Save** on the Manage SSL Auth Private Keys and Certificates screen.

To import a certificate:
1. Click **Import**.
2. Click **Browse** to locate the certificate.
3. Highlight the file and click **Open**.
4. Enter the certificate password.
5. Click **Next**.
6. Click **Done** on the Import Certificate Details screen.
7. Click **Save** on the Manage SSL Auth Private Keys and Certificates screen.

**To view certificate information:**
- Click the certificate Serial number.

**Note:** If a certificate has been revoked, PingFederate indicates this problem in the certificate information window.

**To export a certificate:**
1. Click **Export** under Action for the certificate you want to export.
2. Select **Certificate Only**.
   - Or:
     Select **Certificate and Private Key** and enter an Encryption Password.
3. Click **Next**.
4. On the Certificate Summary screen, click **Export**.
5. Save the file on your system and click **Done**.

**To create a certificate-authority signing request:**
1. Click **Certificate Signing** under Action for the desired certificate.

**Note:** This selection is inactive if you have not yet saved a newly created or imported certificate. Click **Save** and then return to this screen from the Main Menu.

**Tip:** The selection is also inactive if a previously signed certificate has been revoked. Because the revocation may indicate that the private key has been compromised, the best practice is to import or create a replacement certificate for CA signing.

2. Select Generate Certificate Signing Request (CSR), if not already selected.
3. Click **Next**.
4. Click **Generate CSR** on the Generate CSR screen.
5. Click **Next**.
6. On the Certificate Summary screen, click **Export**.
7. Save the file on your system and click **Done**.

**To import a certificate authority response:**
1. Click **Certificate Signing** under Action for the relevant certificate.
2. Select Import CSR Response and click **Next**.
3. Click **Browse** and locate the CSR response to import.
4. Highlight the file and click **Open**.
5. Click **Next**.
6. Click **Done** on the Summary screen.
7. Click **Save** on the Manage SSL Auth Private Keys and Certificates screen.
To delete a certificate:
1. Click **Delete** under **Action** for the certificate you want to delete.

   **Note:** This option is inactive if the certificate is used in any connection or other type of configuration. To enable deletion, click **Check Usage** to locate the configuration(s) that depend on the certificate and then modify each to remove the dependency.

2. Click **Save**.

Create Certificate Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Name</td>
<td>The common name (CN) identifying the certificate.</td>
</tr>
<tr>
<td>Organization</td>
<td>The organization (O) or company name creating the certificate.</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>The specific unit within the organization (OU).</td>
</tr>
<tr>
<td>City</td>
<td>The city or other primary location (L) where the company operates.</td>
</tr>
<tr>
<td>State</td>
<td>The state (ST) or other political unit encompassing the location.</td>
</tr>
<tr>
<td>Country</td>
<td>The country (C) where the company is based.</td>
</tr>
<tr>
<td>Validity (days)</td>
<td>The time during which the certificate is valid.</td>
</tr>
<tr>
<td>Key Algorithm</td>
<td>A mathematical formula used to generate a key. PingFederate uses either of</td>
</tr>
<tr>
<td>(drop-down menu)</td>
<td>two algorithms, RSA or DSA.</td>
</tr>
<tr>
<td>Key Size (bits)</td>
<td>The number of bits used in the key. (RSA-768 to 2048, DSA-768 and 1024)</td>
</tr>
</tbody>
</table>

Digital Signing and Decryption Keys and Certificates

You can use PingFederate to create and maintain your server’s signing certificates, which you may use to sign SAML requests, responses, and assertions. The same type of certificate is also used for XML decryption (see “XML Encryption” on page 30).

**Caution:** For best security, we recommend using separate certificates for signing and decryption.
To create a new certificate:
1. Click Create New.
2. Enter the information on the form.
3. Click Next.
4. On the Summary screen, click Done.
5. Click Save.

To import a certificate:
1. Click Import.
2. Click Browse to locate the certificate.
3. Highlight the file and click Open.
4. Enter the certificate password.
5. Click Next.
6. Click Done.
7. Click Save.

To view certificate information:
- Click the certificate Serial number.

**Note:** If a certificate has been revoked, PingFederate indicates this problem in the certificate information window.

To export a certificate:
1. Click Export under Action for the certificate you want to export.
2. Select Certificate Only.
   - Or:
     - Select Certificate and Private Key and enter an Encryption Password.
3. Click Next.
5. Save the file on your system and click Done.
To create a certificate-authority signing request:

1. Click **Certificate Signing** under Action for the desired certificate.

   **Note:** This selection is inactive if you have not yet saved a newly created or imported certificate. Click **Save** and then return to this screen from the Main Menu.

   **Tip:** The selection is also inactive if a previously signed certificate has been revoked. Because the revocation may indicate that the private key has been compromised, the best practice is to import or create a replacement certificate for CA signing.

2. Select Generate Certificate Signing Request (CSR), if not already selected.
3. Click **Next**.
4. Click **Generate CSR** on the Generate CSR screen.
5. Click **Next**.
6. On the Certificate Summary screen, click **Export**.
7. Save the file on your system and click **Done**.

To import a certificate authority response:

1. Click **Certificate Signing** under Action for the relevant certificate.
2. Select Import CSR Response and click **Next**.
3. Click **Browse** and locate the CSR response to import.
4. Highlight the file and click **Open**.
5. Click **Next**.
6. Click **Done** on the Summary screen.
7. Click **Save** on the Manage Digital Signing Certificates screen.

To delete a certificate:

1. Click **Delete** under Action for the certificate you want to delete.

   **Note:** This option is inactive if the certificate is used in any connection or other type of configuration. To enable deletion, click **Check Usage** to locate the configuration(s) that depend on the certificate and then modify each to remove the dependency.

2. Click **Save**.

### Create Certificate Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Name</td>
<td>The common name (CN) identifying the certificate.</td>
</tr>
<tr>
<td>Organization</td>
<td>The organization (O) or company name creating the certificate.</td>
</tr>
<tr>
<td>Organizational Unit</td>
<td>The specific unit within the organization (OU).</td>
</tr>
<tr>
<td>City</td>
<td>The city or other primary location (L) where the company operates.</td>
</tr>
</tbody>
</table>
Certificate Revocation Checking

By default at runtime, PingFederate attempts to retrieve a CRL to verify that a signing certificate has not been revoked, whenever a CRL distribution-point URL is included within the certificate (see “Certificate Validation” on page 27). Optionally, on the Manage Certificate Revocation screen you can enable and configure OCSP checking as the preferred verification method, depending on your requirements (see “OCSP Revocation Checking” on page 27).

OCSP can be used in place of CRL checking, or CRLs can be retained as a backup method (for failover).

Note: When OCSP is enabled, CRL checking is not done independently—only as a failover option for one or more OCSP failure conditions.

Also on the Manage Certificate Revocation screen, you can change system-default settings for CRL checking, as needed.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>The state (ST) or other political unit encompassing the location.</td>
</tr>
<tr>
<td>Country</td>
<td>The country (C) where the company is based.</td>
</tr>
<tr>
<td>Validity (days)</td>
<td>The time during which the certificate is valid.</td>
</tr>
<tr>
<td>Key Algorithm (drop-down menu)</td>
<td>A mathematical formula used to generate a key. PingFederate uses either of two algorithms, RSA or DSA.</td>
</tr>
<tr>
<td>Key Size (bits)</td>
<td>The number of bits used in the key. (RSA-768 to 2048, DSA-768 and 1024)</td>
</tr>
</tbody>
</table>
To reach this screen:

- On the Main Menu under Security, click **Certificate Revocation Checking**.

**Field Descriptions (For OCSP Checking)**

<table>
<thead>
<tr>
<th>Field/Selection</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable OCSP</td>
<td>Turns on OCSP certificate-revocation checking.</td>
</tr>
<tr>
<td>Default OCSP Responder URL</td>
<td>The location of a URL to use for certificate-revocation checking, a backup used only if the OCSP Responder URL is not contained in the certificate.</td>
</tr>
</tbody>
</table>
## Chapter 5
### Security Management

<table>
<thead>
<tr>
<th>Field/Selection</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default OCSP Responder Signature Verification Certificate</td>
<td>Certificate used to verify that the returned certificate status was sent from the Default OCSP Responder—required if the certificate is not included in the response (click Manage Certificates to import the verification certificate, as needed).</td>
</tr>
<tr>
<td>Do NOT allow Responder to use cached responses</td>
<td>When unchecked (the default), the OCSP Responder uses cached responses when available for the subject certificate (for an indicated period of time—see the description for “Next Update Grace Period,” below). If checked, PingFederate sends a nonce in the request to the Responder, effectively requiring that the status of the certificate be determined in real time. This option is intended to enhance the prevention of Internet replay attacks (in addition to timestamping), where required. <strong>Important:</strong> Making this selection may slow down OCSP response time for a request and will increase general processing overhead at the Responder site.</td>
</tr>
<tr>
<td>This Update Grace Period</td>
<td>For the response to be considered valid, the PingFederate server-clock time must correspond to the &lt;thisUpdate&gt; timestamp in the OCSP response, plus or minus the number of minutes set for this field (to compensate for clock variances).</td>
</tr>
<tr>
<td>Next Update Grace Period</td>
<td>If the response includes a &lt;nextUpdate&gt; timestamp indicating when updated certificate statuses will be available, then PingFederate checks to ensure that the timestamp is not earlier than the current server time, adding this grace period to compensate for clock variances.</td>
</tr>
<tr>
<td>Responder Timeout</td>
<td>The allowable response time before the OCSP Responder URL is considered unavailable and processing continues (see “OCSP Responder is Unavailable,” below).</td>
</tr>
<tr>
<td>Certificate is Unknown</td>
<td>The certificate does not fall under the purview of the CA associated with the OCSP Responder. The drop-down choices indicate whether an unknown certificate is to be considered valid or not, or whether to try CRL checking.</td>
</tr>
<tr>
<td>OCSP Responder is Unavailable</td>
<td>Indicates what action to take if the Responder cannot be reached.</td>
</tr>
<tr>
<td>OCSP Responder Returns Error</td>
<td>Indicates what action to take if the Responder returns an error.</td>
</tr>
<tr>
<td>Proxy Settings</td>
<td>If OCSP messaging is routed through a proxy server, specify the server’s Host (DNS name or IP address) and the Port number. The same proxy information applies to CRL checking, when CRL is enabled for failover.</td>
</tr>
</tbody>
</table>
Field Descriptions (For CRL Checking)

<table>
<thead>
<tr>
<th>Field/Selection</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable CRL Checking</td>
<td>Enables CRL revocation checking (the default). <strong>Note:</strong> CRL checking must remain enabled if any selections for OCSP Error Handling include failover. If OCSP is enabled and no CRL failover is specified, then this selection has no effect.</td>
</tr>
<tr>
<td>Treat Unretrievable CRLs as Revoked</td>
<td>If checked, PingFederate immediately aborts the processing associated with the certificate. If unchecked, the server treats the certificate as valid but continues trying to retrieve the CRL.</td>
</tr>
<tr>
<td>Next Retry on Resolution Failure</td>
<td>Specifies the number of minutes the server waits before trying to retrieve a CRL when the previous attempt failed—applies only when the selection above (Treat Unretrievable CRLs as Revoked) is unchecked.</td>
</tr>
<tr>
<td>Next Retry on Next Update Expiration</td>
<td>How long the server waits before requesting a new CRL when the most recently retrieved CRL (in cache) has a next-update time in the past. <strong>Note:</strong> Certain actions in the administrative console, such as saving changes to an IdP adapter instance, reset the CRL cache. When it happens, PingFederate requests new CRLs for subsequent transactions as needed.</td>
</tr>
<tr>
<td>Verify CRL Signature</td>
<td>When checked (recommended), PingFederate verifies the CRL signature using the public key of the issuer, which must be in the certificate chain or in the list of Trusted CAs (see &quot;Trusted Certificate Authorities&quot; on page 204).</td>
</tr>
<tr>
<td>Proxy Settings</td>
<td>If CRL checking is routed through a proxy server, specify the server’s Host (DNS name or IP address) and the Port number. The same proxy information applies to OCSP checking, when enabled.</td>
</tr>
</tbody>
</table>

Authentication

This portion of the Main Menu, under Server Configuration, allows administrators to manage Basic authentication to the PingFederate server, when needed, as well as authentication needs for secured protocol transactions.
Chapter 5
Security Management

Application Authentication

When you use the SAML 2.0 Attribute Query profile as an SP, password security is required between the application requesting attributes and the SP PingFederate server. Basic authentication is also required for applications making calls to PingFederate’s Connection Management Service and optional for the SSO Directory Service (see “Web Service Interfaces” on page 553).

If you are using the SAML 2.0 Attribute Query profile as an SP, then the requesting application(s) at your site must authenticate to the PingFederate server (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started and “/sp/startAttributeQuery.ping” on page 537).

In addition, authentication is required to access PingFederate runtime data via JMX (see “Runtime Monitoring Using JMX” on page 101) or to make SOAP calls to the Connection Management Web Service. Authentication is optional for the SSO Directory Service (see “Web Service Interfaces” on page 553).

Note: To help ensure network security, access to all of these services is deactivated when PingFederate is first installed.

Administrators can activate and configure authentication for any or all of the services on the Application Authentication screen.

To enable access to a service:
1. Click Activate for the Service under Action.
2. Where required, enter an Id, Shared Secret, and Confirm Shared Secret for the service.
   You and the application developer must agree to these values.
   This step is optional for the SSO Directory Service; the Service can be active without requiring authentication (the default setting).
3. Repeat the steps above for other Services, as needed.
4. Click Save.

To change an application ID or password:
▶ Replace the existing information in the necessary field(s) and click Save.
To block access to an active service:

- Click **Deactivate** for the Service under Action and then click **Save**.

**Tip:** Although not accessible when deactivated, the Connection Management Service and the SSO Directory Service are still deployed by default as part of PingFederate. If your organization does not plan to use one (or either) of these services, you may wish to remove the corresponding WAR file(s) from the `<pf_install>/pingfederate/server/deploy` or (and) `../deploy2` directories, respectively:

```
pf-ws.war
pf-mgmt-ws.war
```

### Validating Password Credentials

PingFederate provides an authentication mechanism using plug-in Password Credential Validators. This feature provides centralized credential validation for other PingFederate configurations. Currently, instances of credential validators are used when configuring an HTTP Basic or HTML Form IdP Adapter (see “Configuring the HTTP Basic IdP Adapter” on page 509 and “Configuring the HTML Form IdP Adapter” on page 515) and for OAuth resource-owner grant configurations (see “Grant Types” on page 11 and “Resource-Owner Credentials Mapping” on page 181).

PingFederate is distributed with the following plug-in Password Credential Validators:

- **LDAP Username/Password** – Validates credentials based on an LDAP look-up in an organization's user-data store.
- **Simple Username/Password** – Validates credentials maintained by PingFederate.
- **RADIUS Username/Password** – Validates credentials based on the RADIUS protocol on an organization's RADIUS server.

To configure an instance of a credentials validator, click **Create New Instance**.

To edit an existing instance, click its Instance Name.

**To delete an existing instance:**

1. Click **Delete** next to the Instance Name (To undo the deletion, click **Undelete**.)
2. Click **Save** to confirm the deletion.

### Choosing a Type

The first step in this configuration is choosing a credentials-validator type. Available types are determined by plug-in JAR files loaded in the `<pf_install>/pingfederate/server/default/deploy` directory. Several validator plug-ins are bundled with PingFederate. Other plug-ins may be added periodically, available from Ping Identity (pingidentity.com/support-and-downloads).

![Create Credential Validator Instance](image)

#### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Name</td>
<td>A descriptive name for the credentials-validator instance—for example, an identity management system name.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>An internal identifier for the credentials-validator instance. Must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Type</td>
<td>A list of deployed credentials-validator types available for creating a credentials-validator instance for the server. A developer typically deploys any new credentials- types before an administrator sets up a connection partner.</td>
</tr>
<tr>
<td>Parent Instance (Optional)</td>
<td>A list of configured instances for this type of credentials-validator. If applicable, select an instance that can be used as a basis for a parent/child configuration.</td>
</tr>
</tbody>
</table>

**To specify a validator instance:**

1. Enter the Instance Name and Instance Id on the Type screen.
2. Select the Type from the drop-down menu. If the validator type you need is not listed, click Visit PingIdentity.com for additional types to see if a suitable validator plug-in is available from the download site.

3. (Optional) Select a Parent Instance from the drop-down list. If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

4. Click Next and enter information on the configuration screen for this validator instance. This configuration varies depending on the validators deployed on your server. For add-on validators please consult the online documentation referenced in the download package, or look under Product Documentation at pingidentity.com. For validators bundled with PingFederate, refer to one of the following sections:
   - “Configuring the LDAP Credential Validator”, next
   - “Configuring the Simple Credential Validator” on page 221
   - “Configuring the RADIUS Credential Validator” on page 222

**Configuring the LDAP Credential Validator**

The LDAP Username/Password Credential Validator verifies credentials using an organization’s user-data store.

In addition to lookup configuration, this screen provides a means of defining error-message interpretation for LDAP stores other than Microsoft Active Directory or Oracle Directory Server. (Message parsing for AD and Oracle messages is built into PingFederate; however, an administrator may also override the default message handling for those data stores on this screen.)

*Tip:* LDAP server messages are used by the HTML Form Adapter to determine LDAP password-change scenarios and then present relevant messages to end users (see “Configuring the HTML Form IdP Adapter” on page 515). The end-user messages are configurable for the associated HTML templates and may be localized (see “Customizing User-Facing Screens” on page 87).
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAP Datastore</td>
<td>The LDAP data store configured in PingFederate. If you have not yet configured the server to communicate with the LDAP server you need, click <strong>Manage Data Stores</strong>. <strong>Note:</strong> If you are using Active Directory and want to support password changes (see “Configuring the HTML Form IdP Adapter” on page 515), ensure that the connection to Active Directory is secured using LDAPS. Active Directory requires this level of security to allow password changes (see “Configuring an LDAP Connection” on page 121).</td>
</tr>
<tr>
<td>Search Base</td>
<td>The location in the LDAP directory server from which the search begins.</td>
</tr>
<tr>
<td>Search Filter</td>
<td>Query used to produce the desired set of matching records.</td>
</tr>
<tr>
<td>Scope of Search</td>
<td>The level of search to be performed in the search base. One level indicates a search of objects immediately subordinate to the base object, not including the base object itself. Subtree indicates a search of the base object and the entire subtree within the base object distinguished name.</td>
</tr>
</tbody>
</table>
To complete the LDAP validation:

1. (Optional) Click Add a new row to ‘Authentication Error Overrides’.

   **Tip:** This option may be required for an LDAP directory other than AD or Oracle to support the HTML Form Adapter’s password reset function or to alter end-user messages associated with that function (see “HTML Form Adapter Configuration” on page 515).

   a. Enter an expression, using wildcard asterisks (*), to match against messages returned from the LDAP server.
      For example: *expired*
   b. Select a relevant error condition from the Error drop-down list.
   c. Click Update under Action.
   d. Repeat these steps for additional overrides as needed.

   After overrides are configured, you can use the links under Action to edit or delete an override, or to change their display order. (The ordering does not affect runtime processing.)

2. Select the LDAP Datastore and enter information into the required fields, as described under Field Descriptions above.

**To edit an authentication error override:**

1. Click Edit next to the authentication error override.
2. Change information as needed, then click Update.

**To delete a authentication error override:**

   ▶ Click Delete next to the authentication error override.

**Configuring the Simple Credential Validator**

The Simple Username/Password Credential Validator verifies credentials maintained by PingFederate.

**Note:** This validator is best used for testing purposes or for an organization with few accounts.
To add users:
1. Click Add a new row to ‘Users’.
2. Enter a Username and Password and click Update.
3. Repeat the previous steps for additional users as needed.

To edit user credentials:
1. Click Edit next to the user’s credentials.
2. Change information as needed, then click Update.

To delete a user:
- Click Delete next to the user’s credentials.

Configuring the RADIUS Credential Validator

The RADIUS Username/Password Credential Validator verifies credentials using the RADIUS protocol.

RADIUS supports strong authentication with both one-step (a combination of regular password and a one-time password in one field) and two-step (challenge-response) authentication. Two-step authentication is supported in the HTML Form IdP Adapter.

**Tip:** RADIUS server messages are used by the HTML Form Adapter to determine two-step authentication scenarios and then present a logon screen to end users. The screen is configurable and may be localized (see “Customizing User-Facing Screens” on page 87).

---

**Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hostname</td>
<td>The IP address of the RADIUS server. For failover, you can enter one or more backup RADIUS servers by adding each server in its own row of the table. Each row represents a distinct RADIUS server that can be used for failover. PingFederate attempts to make a connection to each server in the order listed until a successful connection is obtained. If authentication to this Password Credential Validator fails, control is returned to the adapter using it so that other Password Credential Validators can be tried (if defined).</td>
</tr>
<tr>
<td>Authentication Port</td>
<td>The UDP (User Datagram Protocol) port used to authenticate to the RADIUS server. The default value is 1812.</td>
</tr>
<tr>
<td>Authentication Protocol</td>
<td>The protocol used to authenticate to the RADIUS server. The available choices are Password Authentication Protocol (PAP) and Challenge Handshake Authentication Protocol (CHAP). Select the protocol expected by your RADIUS server. The default value is PAP.</td>
</tr>
<tr>
<td>Shared Secret</td>
<td>The password shared between PingFederate and the RADIUS server used to encrypt passwords.</td>
</tr>
</tbody>
</table>

Note: The NAS-IP-Address attribute is added to all Access-Request packets sent to the RADIUS server. The value is copied from the pf.engine.bind.address property in run.properties. Only IPv4 addresses are supported.

To complete the RADIUS validation:
1. Click Add a new row to ‘RADIUS Servers’.
2. Enter a Hostname, Authentication Port, and Shared Secret and click Update.
3. Repeat the previous steps for additional servers as needed.

To edit a RADIUS server:
1. Click Edit next to the server’s credentials.
2. Change information as needed, then click Update.

To delete a RADIUS server:
1. Click Delete next to the RADIUS server.
Setting Advanced Options

► (Optional) Click Show Advanced Fields to reconfigure default settings for the RADIUS instance, as needed.

Refer to the on-screen field descriptions and the following table for more information.

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAS Identifier</td>
<td>(Required) The attribute identifying the NAS (Network Access Server)</td>
</tr>
<tr>
<td></td>
<td>originating the request for access. The default value is PingFederate.</td>
</tr>
<tr>
<td>Timeout</td>
<td>(Required) The maximum number of milliseconds before a connection timeout</td>
</tr>
<tr>
<td></td>
<td>to the RADIUS server.</td>
</tr>
<tr>
<td>Retry Count</td>
<td>(Required) Number of times to retry a failed connection before moving to</td>
</tr>
<tr>
<td></td>
<td>the next host.</td>
</tr>
</tbody>
</table>

Extending Contracts for the RADIUS Credential Validator

In some cases, you might want to extend contracts in the RADIUS Password Credential Validator. For example, you might use extended attributes to map into a USER_KEY for an OAuth persistent grant configuration (see “About OAuth” on page 10 and “Resource-Owner Credentials Mapping” on page 181).

Tip: If you are configuring the HTML Form IdP Adapter with the RADIUS credential validator, you only need to extend the contract of the adapter itself. (see “Configuring the HTML Form IdP Adapter” on page 515).

Vendor specific RADIUS attributes can be made available by extending the RADIUS attribute dictionary. Copy the vendor-specific attribute dictionaries into the pingfederate/server/default/conf/radius folder. The format of the dictionaries must use the FreeRADIUS dictionary syntax. Then edit the existing dictionary file to include each of them.

To extend contracts for a RADIUS credential validator:

► (Optional) On the Extended Contract screen, click Add. Continue to add attributes, as needed.
Finishing the Validator Configuration

To complete and save the configuration, click **Done** on the Summary screen and then **Save** on the Manage Credential Validators screen.

Using AD Domains and Kerberos Realms

Active Directory (AD) Domains and Kerberos Realms provide PingFederate with a centralized configuration for verifying authenticated users via adapters or token processors, including:

- **PingFederate Integrated Windows Authentication (IWA) Integration Kit** (version 3.0 and later) – Using the IWA Adapter with a configured AD Domain allows a PingFederate IdP server to perform SSO to SP applications based on IWA credentials.
- **PingFederate Kerberos Token Processor** (version 2.0 and later) – This token processor allows the PingFederate WS-Trust Security Token Service to accept and validate Kerberos tokens—via a configured Kerberos Realm—from a Web Service Client (see “Token Processors and Generators” on page 7).

**Tip:** Find User Guides and other documentation for Integration Kits and Token Processors under Product Documentation at pingidentity.com (documentation.pingidentity.com/display/LP/Product+Documentation).

To configure an AD Domain or Kerberos Realm, click **Add Domain/Realm**.

To edit an existing Domain or Realm, click its name.

Configuring a Domain or Realm

Use this screen to configure AD Domains or Kerberos Realms that PingFederate can use to contact Domain Controller(s) or Key Distribution Center(s) for verifying user authentication.
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain/Realm Name</td>
<td>The fully-qualified domain or realm name. For example: corporation.companydomain.com</td>
</tr>
<tr>
<td>Domain/Realm Username</td>
<td>The ID for the domain or realm account name.</td>
</tr>
<tr>
<td>Domain/Realm Password</td>
<td>The password for the domain or realm account.</td>
</tr>
<tr>
<td>Domain Controller/Key Distribution Center Host Names</td>
<td>(Optional) Specify the host name or IP address for the domain controller or KDC and click <strong>Add</strong>. For example: fn_dc1. If unspecified, PingFederate uses a DNS lookup.</td>
</tr>
</tbody>
</table>
### Authentication

**Managing Domain or Realm Settings**

Use this screen to change default security and logging settings for all configured AD Domains and Kerberos Realms.

<table>
<thead>
<tr>
<th>Field Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Domain/Realm Connectivity (button)</td>
<td>Tests access to the domain controller or KDC from the administrative-console server. When a connection to any of the configured controllers/KDCs is successful, the message <strong>Test Successful</strong> appears. Otherwise, the test returns error messages near the top of the screen. For multiple connections, note that the test stops at the first successful result, so all connections are not necessarily verified. Also, connectivity may be subsequently affected in different deployment scenarios, including for engine server nodes running in a clustered environment. <strong>Tip:</strong> For help resolving connection issues, select the <strong>Debug Log Output</strong> checkbox on the Manage Domain/Realm Settings screen, and run the test again to view the debug output to the PingFederate server.log (see Managing Domain or Realm Settings (the next section). For server clusters, you can also push this setting to PingFederate engine nodes as needed (see the Server Clustering Guide).</td>
</tr>
<tr>
<td>Force TCP</td>
<td>When selected, requires use of the Transmission Control Protocol instead of the default User Datagram Protocol. Use this option when firewall or network configurations require acknowledgement that packets are properly received. <strong>Note:</strong> If you choose this option, ensure that you restart PingFederate after saving the configuration.</td>
</tr>
</tbody>
</table>

AD Domain Controller/Key Distribution Center Timeout (secs) | 3 |

AD Domain Controller/Key Distribution Center Retries | 3 |
# Security Management

## Debug Log Output

When selected, sends verbose debugging output to the PingFederate `server.log` for all interactions with Domain Controllers or Key Distribution Centers (KDCs). For more information on the `server.log`, see “Managing Log Files” on page 44.

## AD Domain Controller/Key Distribution Center Timeout (secs)

Specifies the amount of time (in seconds) PingFederate waits for a network response from a domain controller or KDC. The default is 3 seconds.

**Note:** This value applies to each attempt PingFederate makes to contact the domain controller or KDC.

**Note:** The new timeout takes effect only after PingFederate is restarted, after you save the configuration.

## AD Domain Controller/Key Distribution Center Retries

Specifies the number of times PingFederate tries contacting the domain controller or KDC. The default is 3 times.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug Log Output</td>
<td>When selected, sends verbose debugging output to the PingFederate <code>server.log</code> for all interactions with Domain Controllers or Key Distribution Centers (KDCs). For more information on the <code>server.log</code>, see “Managing Log Files” on page 44.</td>
</tr>
<tr>
<td>AD Domain Controller/Key Distribution Center Timeout (secs)</td>
<td>Specifies the amount of time (in seconds) PingFederate waits for a network response from a domain controller or KDC. The default is 3 seconds. <strong>Note:</strong> This value applies to each attempt PingFederate makes to contact the domain controller or KDC. <strong>Note:</strong> The new timeout takes effect only after PingFederate is restarted, after you save the configuration.</td>
</tr>
<tr>
<td>AD Domain Controller/Key Distribution Center Retries</td>
<td>Specifies the number of times PingFederate tries contacting the domain controller or KDC. The default is 3 times.</td>
</tr>
</tbody>
</table>
Identity Provider SSO Configuration

In an IdP role, you use the PingFederate administrative console to configure local application-integration information and to manage connections to your SP-partner sites. You must configure Server Settings from the Main Menu to establish your site as an IdP before configuring connections to SPs (see “Choosing Roles and Protocols” on page 106).

Note that only one connection is needed per partner, even if you are targeting more than one Web application at the destination SP site. You can configure more than one connection, however, if your partner supports multiple protocols, or supports multiple federation IDs for the same protocol (see “Federation Server Identification” on page 37).

Note: This chapter applies to configuration settings needed for browser-based SSO. While there is some cross-over information also applicable to WS-Trust STS, if you are using PingFederate exclusively as an STS, start with “WS-Trust STS Configuration” on page 437.

Under some conditions, you can enable SSO for an unlimited number of partners at once by configuring a single, common connection (see “Using Auto-Connect” on page 30).
Chapter 6
Identity Provider SSO Configuration

Application Integration Settings

The integration of local applications with PingFederate is the essential “first-mile” configuration that allows end-users to access protected resources across domains. This process is facilitated through the use of application-integration kits and a robust Software Development Kit (see “SSO Integration Kits and Adapters” on page 14).

Under Application Integration Settings on the Main Menu, you configure the IdP Adapters that PingFederate needs to interact with applications or access-management systems used to authenticate users at your site. You can also set a Default URL to which users may be directed during SLO, and you can look up system endpoints that application developers at your site need to access PingFederate’s SSO/SLO services.

Note: If your PingFederate configuration enables the WS-Trust STS, the selections under Application Integration Settings also include links for configuring plug-in Token Processors and, optionally, STS Request Parameters. Locate configuration information under “IdP Configuration for STS” on page 443.

Configuring IdP Adapters

An IdP adapter is used to look up session information and provide user identification to PingFederate (see “SSO Integration Kits and Adapters” on page 14).

You must configure at least one instance of an IdP adapter in order to set up connections to SP partners.

For information about configuring the adapters packaged with PingFederate, see:

- “Configuring the IdP OpenToken Adapter” on page 500
- “Configuring the HTTP Basic IdP Adapter” on page 509
- “Configuring the HTML Form IdP Adapter” on page 515
- “Configuring the Composite Adapter” on page 523

You reach this screen by clicking Adapters under Application Integration Settings in IdP Configuration.

To configure a new instance:
- Click Create New Instance.

To edit an existing adapter instance:
- Click the Instance Name and click the step you need to change.
To delete an adapter instance:
1. Click **Delete** next to the Instance Name. (To undo the deletion, click **Undelete**.)

   **Note:** This option is inactive if the instance is referenced elsewhere, or if it is a parent to other instances. To enable deletion, click **Check Usage** to locate the configuration(s) that depend on the instance and go to each listed configuration to remove the dependencies. If the instance you want to delete is a parent, delete child instances first.

2. Click **Save** to confirm the deletion.

**Selecting an IdP Adapter Type**

The first step in creating an adapter instance is choosing an adapter type. Available adapter types are determined by JAR files loaded in the `<pf_install>/pingfederate/server/default/deploy` directory. Some adapters are bundled with PingFederate (see “SSO Integration Kits and Adapters” on page 14). Other adapters and integration kits are available from the Ping Identity Web site (www.pingidentity.com/support-and-downloads).

**Field Descriptions**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Name</td>
<td>A descriptive name for the adapter instance—for example, an identity management system name.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>An internal identifier for the adapter instance. Must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Type</td>
<td>A list of deployed adapter types available for creating an adapter instance for the server. A developer typically deploys any new adapter types before an administrator sets up a connection partner.</td>
</tr>
<tr>
<td>Parent Instance (Optional)</td>
<td>A list of configured instances for this type of adapter. If applicable, select an instance that can be used as a basis for a parent/child configuration.</td>
</tr>
</tbody>
</table>
To define an adapter instance:
1. Enter the Instance Name and Instance Id on the Type screen.
2. Select the Type from the drop-down menu.
   If the adapter you need is not listed, click **Visit PingIdentity.com for additional types** to see if a suitable adapter is available from the PingFederate download site, or create your own adapter (see “SSO Integration Kits and Adapters” on page 14).
3. (Optional) Select a Parent Instance from the drop-down list.
   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
4. Click Next and enter information on subsequent screens for this adapter setup.
5. Click Done on the Adapter Summary screen.
6. Click Save on the Manage IdP Adapter Instances screen.

**Tip:** The setup steps and information needed at those steps vary with the adapters deployed on your server (see “SSO Integration Kits and Adapters” on page 14). For information about configuring the adapters packaged with PingFederate, see “OpenToken Adapter Configuration” on page 499, “Configuring the HTTP Basic IdP Adapter” on page 509, “Configuring the HTML Form IdP Adapter” on page 515, or “Configuring the Composite Adapter” on page 523.

5. Click Done on the Adapter Summary screen.
6. Click Save on the Manage IdP Adapter Instances screen.

**Configuring an IdP Adapter**

Depending on the adapter you choose, different configuration parameters are available on the IdP Adapter screen. These options are controlled by the adapter plug-in software (see “SSO Integration Kits and Adapters” on page 14).

- For information about configuring the OpenToken Adapter, see “OpenToken Adapter Configuration” on page 499.
- For information about configuring the HTTP Basic IdP Adapter, see “Configuring the HTTP Basic IdP Adapter” on page 509.
- For information about configuring the HTML Form IdP Adapter, see “Configuring the HTML Form IdP Adapter” on page 515.
- For information about configuring the Composite Adapter, see “Configuring the Composite Adapter” on page 523.
- For information about configuring an adapter contained in an integration kit, locate the User Guide under Product Documentation at pingidentity.com.

**Important:** If you change adapters that are used by existing partner connections, you may need to reconfigure those connections. If so, a **Fix Errors** link appears on the Manage IdP Adapter Instances screen. Click the link to navigate to the screens you need to reconfigure. You cannot save the changes to the adapter until the existing connections have been repaired.
Invoking Adapter Actions

Adapters may be written to provide configuration assistance or validation actions. Actions may also include generation of parameters that might need to be set manually in a configuration file.

For information about actions available using the OpenToken Adapter, see “Configuring the IdP OpenToken Adapter” on page 500.

To reach this screen:

1. Click Adapters on the Main Menu.
2. Click an Instance Name on the Manage IdP Adapter Instances screen.
3. Click Actions (if available).

To generate a properties list:

- Click Download under Action Invocation Link.

Extending an Adapter Contract

Adapters may be written with an option allowing administrators to add to the attributes that the adapter returns from a user’s session. The PingFederate OpenToken Adapter, for example, provides such an option (see “OpenToken Adapter Configuration” on page 499).

Note: For the Composite Adapter, attributes for IdP Adapter Instances that comprise the composite configuration must be added on this screen (see “Configuring the Composite Adapter” on page 523).

To reach this screen:

- Click Adapters on the Main Menu.
- Click an Instance Name.
- Click Extended Contract (if available).

Note: If this is a child instance, select the override checkbox to modify the configuration.
To add an attribute:
▶ Enter the attribute name in the text box and click Add.

Setting Pseudonym Values and Masking

On the Adapter Attributes screen you must select attributes to use for generating a pseudonym identifier (see “Account Linking” on page 17).

Optionally on this screen, you can also choose to mask the values of any or all attributes that PingFederate logs from this adapter instance at runtime (see “Attribute Masking” on page 23).

To configure Pseudonym generation:
▶ Under Pseudonym select the value(s) to use.

Note: A selection is required regardless of whether you use pseudonyms for account linking. This allows account linking to be used later without having to delete and reconfigure the adapter. Ensure that you choose at least one attribute that is unique for each user (for example, email) to prevent the same pseudonym from being assigned to multiple users.

To mask attributes in log files:
▶ Under Mask Log Values select the attribute(s) whose value(s) you want to mask.

If OGNL expressions might be used to map derived values into outgoing assertions and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

To reach this screen:
1. Click Adapters on the Main Menu.
2. Click an Instance Name.
3. Click Adapter Attributes.

Selecting an Authentication Context

If you have deployed an integration kit that supports authentication context, you can specify the context in the ldP adapter configuration under Advanced Fields.

(For a discussion of authentication context, find that term under “Terminology” in the “Supported Standards” chapter of Getting Started. For detailed information, see the OASIS SAML document saml-authn-context-2.0-os.pdf.)
To enter an authentication context URI for an adapter that supports this feature, click **Advanced Fields** on the adapter configuration screen.

**Editing and Saving Adapter Instances**

From the Adapter Instance Summary screen, you can reach adapter settings for editing.

**To edit the configuration:**
1. Click the heading above the information you want to change.
2. Make your changes.
3. Click **Save** on the configuration page and on the Manage IdP Adapter Instances screen.

**To save an adapter instance:**
1. Click **Done** on the Summary screen.
2. Click **Save** on the Manage IdP Adapter Instances screen.

**Configuring Adapter Selectors**

Adapter selectors provide a plug-in capability for PingFederate to choose among configured IdP adapter instances for any SSO request and some OAuth flows that involve the browser. PingFederate chooses adapter instances based on criteria you define in a configured adapter selector or series of selectors. This optional feature provides for global and flexible authentication-adapter usage across SP connections.

**Important:** Before configuring adapter selectors, it is best to configure the IdP Adapters you want to use (see “Configuring IdP Adapters” on page 230). Also, to be used, the same instances of those adapters must be configured for attribute mapping in applicable SP connections (see “IdP Adapter Mapping” on page 265).

The configuration of plug-in selectors packaged in this PingFederate release—see “Bundled IdP Adapter Selectors” on page 15—is described in the following sections. For information on the configuration of other plug-ins, please consult the associated documentation.

- Reach this screen by clicking **Adapter Selection** under Application Integration Settings in IdP Configuration.
- To configure an instance of an adapter selector, click **Create New Instance**.
- To edit an existing instance, click its **Instance Name**.
To delete an instance:
1. Click Delete next to the Instance Name. (To undo the deletion, click Undelete.)
2. Click Save to confirm the deletion.

Choosing a Selector Type

On this screen an administrator chooses the adapter selector to configure. Many selector plug-ins are bundled with PingFederate (see “Bundled IdP Adapter Selectors” on page 15). Other plug-ins may be added periodically, available from Ping Identity (pingidentity.com/support-and-downloads).

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Name</td>
<td>A descriptive name for the selector instance.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>An internal identifier for the selector instance—must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Type</td>
<td>A list of deployed selector types available for creating an instance for the server.</td>
</tr>
</tbody>
</table>

To specify a selector instance:
1. Enter the Instance Name and Instance Id on the Type screen.
2. Select the Type from the drop-down list.
3. Click Next and enter information on the configuration screen for this adapter-selector instance.

This configuration varies depending on the selectors deployed on your server. For selectors bundled with PingFederate, refer to one of the following sections:

- “Configuring the CIDR Adapter Selector” on page 237
- “Configuring the Cluster Node Selector” on page 237
- “Configuring the Connection Set Selector” on page 238
- “Configuring the HTTP Header Selector” on page 238
- “Configuring the OAuth Scope Selector” on page 239
- “Configuring the Requested AuthN Context Selector” on page 240
Configuring the CIDR Adapter Selector

This adapter selector determines which IdP adapter instance to use based on the IP address of an incoming SSO request.

To configure the selector:

1. Under Action, click Add a new row to ‘Networks’.
2. Enter a Network Range (IPv4 addresses only) and click Update.

**Note:** If you want to include all IP addresses for testing, add two separate ranges: 0.0.0.0/1 and 128.0.0.0/1. The CIDR Adapter Selector interprets a specification of 0.0.0.0/0 as an empty range rather than as a wildcard for all addresses.

3. Repeat the previous step as needed for additional ranges.
4. (Optional) Enter a value for Result Attribute Name.

   This field provides a means to indicate in the SAML assertion that a network range was matched during processing (values: Yes or No). Any IdP adapters configured as a result of this adapter selector must have their attribute contract extended with the Result Attribute Name’s field value in order to use its value to fulfill an attribute contract or for issuance criteria. For more information about adapter contract, see “Extending an Adapter Contract” on page 233.

Configuring the Cluster Node Selector

This adapter selector enables PingFederate to choose configured IdP adapters based on the PingFederate cluster node that is servicing the request (for more information on clustering, see the PingFederate Server Clustering Guide). For example, this selector allows you to choose whether or not Integrated Windows Authentication is attempted based on the PingFederate cluster node with which a Key Distribution Center is associated.

On the Adapter Selector screen for this selector, there are no individual configurable fields. Click Next to continue to the Selector Result Values screen to define cluster node index numbers for specific adapter instances (see “Defining Cluster Node Results” next).

Defining Cluster Node Results

On the Selector Result Values screen, list the index numbers for each of the cluster nodes to be associated with specific adapter instances (see “Mapping Selector Results to Adapter Instances” on page 242). These results are used as the criteria for adapter selection.
To add values:
1. Enter a node index number as the Result Value and click Add.
2. Add more index numbers to differentiate criteria for adapter selection.

Configuring the Connection Set Selector

This adapter selector enables PingFederate to choose configured IdP adapters based on a match found between the target SP connection used in an SSO request and SP connections configured within PingFederate. This selector allows you to override connection adapter selection on an individual connection basis.

To configure the selector:
1. Under Action, click Add a new row to ‘Connections’.
2. Select an SP connection from the drop-down list and click Update.
   At runtime, the selector compares the target SP connection in the SSO request to SP connections configured here. If a match is found, the selector returns a result value of Yes.
3. Repeat the previous step as needed for additional connections.
   There is no priority given to the order in which the connections appear.

Configuring the HTTP Header Selector

This adapter selector enables PingFederate to choose configured IdP adapters based on a match found in a specified HTTP header. For example, use an instance of this selector to choose an IdP adapter based on the end user's browser identified by the User-Agent HTTP header.
Application Integration Settings

To configure the selector:

1. Under Action, click Add a new row to ‘Results’.
2. Enter an expression for use when inspecting the HTTP header value of the target HTTP header and click Update.

   This field is case-sensitive and wildcard entries are allowed—for example, *Firefox*

   At runtime, the selector compares the HTTP header to these wildcard expressions. If a match is found, the selector returns a result value of Yes.

   The following expressions work to identify these commonly used browsers.

<table>
<thead>
<tr>
<th>Browser</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td><em>Chrome</em></td>
</tr>
<tr>
<td>Safari</td>
<td><em>Safari</em></td>
</tr>
<tr>
<td>Firefox</td>
<td><em>Firefox</em></td>
</tr>
<tr>
<td>Internet Explorer</td>
<td><em>MSIE</em></td>
</tr>
</tbody>
</table>

3. Repeat the previous step as needed to add more wildcard expressions.
4. Enter the Header Name for the type of HTTP Header you want the selector to inspect — for example, User-Agent. This field is not case-sensitive.

Configuring the OAuth Scope Selector

This adapter selector enables PingFederate to choose configured IdP adapters based on a match found between the scopes of an OAuth authorization request and scopes configured in the PingFederate OAuth Authorization Server (AS). This selector allows you to control the strength of authentication based on client access requirements. For example, if a client requires write access to a resource, you can configure the selector to choose an adapter that offers a stronger form of
authentication such as the X.509 client certificate rather than username and password.

**Note:** To configure an OAuth Scope Selector, you must first configure scopes for the PingFederate OAuth AS (see "Authorization Server Settings" on page 159).

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>FIELD VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMIN</td>
<td></td>
<td>admin permissions</td>
</tr>
<tr>
<td>CREATE</td>
<td></td>
<td>create permissions</td>
</tr>
<tr>
<td>DELETE</td>
<td></td>
<td>delete permissions</td>
</tr>
<tr>
<td>READ</td>
<td></td>
<td>read permissions</td>
</tr>
<tr>
<td>UPDATE</td>
<td></td>
<td>update permissions</td>
</tr>
</tbody>
</table>

**To configure the selector:**

- Select the required scopes and click **Next**.

  At runtime, the selector compares the requested OAuth scopes to the scopes selected here. All of the selected scopes must match for the selector to return a result value of **Yes**.

**Note:** This selector matches only scopes from OAuth authorization requests to the `/as/authorization.oauth2` endpoint. SAML SSO requests will not match the adapter selector’s criteria and result in a returned result value of **No**. Therefore, if you are using this selector as well as selectors specific to SAML connections, place this selector first in the mapping list so that it takes precedence for OAuth without disrupting selector logic on SAML connections (see "Mapping Selector Results to Adapter Instances” on page 242).

**Configuring the Requested AuthN Context Selector**

This adapter selector enables PingFederate to choose configured IdP adapters based on the authentication context requested by an SP partner. The Adapter Selector screen provides an option to use the authentication-context selection result in SAML assertions. (Context results are defined on the next screen—see “Defining AuthN Context Results” on page 241.)

**Note:** This adapter selector works in conjunction with SP partner connections via SAML 2.0 only, using SP-initiated SSO. Other federation protocols do not support authentication context.
When selected (the default), the checkbox on this screen provides a means of either:

- Adding the value of the authentication context determined by the selector into the SAML assertion; or,
- When applicable, replacing any value returned from the associated adapter instance with the selector-result value.

Adapter instances are mapped to the results of selector instances in a subsequent screen setup—see “Mapping Selector Results to Adapter Instances” on page 242.

**Defining AuthN Context Results**

On the Selector Result Values screen, list the authentication contexts to be associated with specific adapter instances (see “Mapping Selector Results to Adapter Instances” on page 242). These results are also used as the criteria for adapter selection.

**To add values:**

1. Enter a Result Value and click **Add**.

   Values may include URIs defined in the SAML 2.0 specifications (see the OASIS SAML document [saml-authn-context-2.0-os.pdf](#)) or any other value agreed upon with an SP partner.

2. Add more values to differentiate criteria for adapter selection.

**Finishing the Selector-Instance Configuration**

To complete the selector-instance configuration, click **Done** on the Summary screen.

- On the Manage Adapter Selector Instances screen, add more instances or click **Next** to complete the selector configuration (see the next section, “Mapping Selector Results to Adapter Instances”).

⚠️ **Caution:** If you are yet not ready to map results, be sure to click **Save** on the Manage Adapter Selector Instances screen. Selector-instance configurations are not retained until you click **Save** either on this screen or on the Map Results to Adapter Instances screen.
Mapping Selector Results to Adapter Instances

An administrator can map the results of selector instances either to adapter instances or to other selectors for subsequent evaluation of additional criteria (for example, to determine whether off-site SSO requests are also authenticated in a certain context or originating from certain browser types).

**Note:** At runtime, the order of adapter-instance selection attempts is determined by the order of selector-instance mappings on this screen. The selection process stops when a match is found, and control is passed to the SP connection runtime configuration. If the match is not allowed based on restrictions imposed (see “Restricting an IdP Adapter to certain Virtual Server IDs” on page 269), PingFederate falls back on the Default Adapter (if any) or selects among the available adapter instances configured for the applicable SP connection (see Step 2 and Step 7). Change the order as needed using the arrows in the left column.

---

**To complete this configuration:**

1. Ensure the checkbox Enable Adapter Selection is selected.

   **Tip:** This checkbox allows an administrator to turn adapter selection on and off and is primarily useful during deployment testing. If this checkbox is *not* selected, all adapter-selection mappings are deactivated and adapter usage is based only on instances configured within SP connections.

2. (Optional) Select the checkbox Fail If No Selection.

   If no adapter instances match either the selector-result mappings or any Default Adapters listed at the bottom of the screen (see Step 7), PingFederate selects among the adapter instances configured for the applicable SP connection. This checkbox overrides this behavior, allowing for cases where strict authentication policy might be required. When selected, an SSO request returns a failure.
message if no adapter instances satisfy requirements specified in this configuration.

3. Click the Selector Instance Name drop-down and choose an instance that needs mapping.

4. Under Instance for Selector Result Values, choose either an adapter or a subsequent selector for the values from the drop-down lists.

   **Note:** Mapping all values is not required: for example, you might leave “None” selected if you want the result to continue down to the next “decision tree.”

   The default Yes/No values for selector instances indicate whether the pre-authentication circumstances meet the defined condition or not.

   If an applicable adapter instance is not listed, you can create it—click **Manage IdP Adapters**.

5. If you chose a selector in the previous step for any result value, repeat that step for the new Selector Result Values and Instance columns that appear at the right.

   **Note:** Repeat this step for subsequent selectors shown in the rightmost Instance column.

6. Repeat Step 3 through Step 5 for each Selector Instance Name that needs mapping.

7. (Optional) Under Default Adapters, choose an adapter instance from the drop-down list.

   Repeat this step to choose additional default instances, if needed.

   If no adapter instances match any of the selector mappings at runtime, the adapter instances listed in this column are tried in order. The first instance that is mapped in the target SP connection configuration is the one used.

8. Click Save.

### Configuring a Default URL and Error Message

As an IdP, you can specify to prompt end users to confirm their single logout requests and a default URL indicating a successful SLO to the end-user (if no other page is designated). On the IdP Default URL page, you can also customize an error message to be displayed as part of the error page rendered in the end-user’s browser if an error occurs during IdP-initiated SSO. For example, you might consider modifying the
default text to include useful information regarding whom the user should contact or what their next step should be.

**Note:** The error message is displayed only when the application calling the start-SSO endpoint does not explicitly provide its own error page URL. The default entry in this field is used to localize the message. For information about how to find and change the default English message and how to use the PingFederate localization feature, see “Localization” on page 92. If localization is not needed, you may also specify a message directly in this field to change the default.

Your application or your partner’s application may supply the URL at runtime (see “IdP Endpoints” on page 529), but if none is provided, PingFederate will use the default value you enter on this screen.

**Tip:** If you leave the default URL blank, PingFederate provides built-in landing page for the user. This Web page is among the templates you can modify with your own branding or other information (see “Customizing User-Facing Screens” on page 87).

### Viewing Application Endpoints

Click **Application Endpoints** on the Main Menu to see a list of endpoints and descriptions applicable to your federation role (IdP or SP). These endpoints are built into PingFederate and cannot be changed.

Web-application developers at your site need to know the application endpoints to initiate transactions via PingFederate (see “SSO Integration Kits and Adapters” on page 14).

**Note:** For specific parameters required or allowed for Application Endpoints, see “IdP Endpoints” on page 529.

This screen also shows a Maintenance Endpoint that you can use to verify that the PingFederate server is running (see “System-Services Endpoints” on page 540).
Viewing Protocol Endpoints

Click **Protocol Endpoints** under Federation Settings in the IdP Configuration section of the Main Menu to see a list of SAML, WS-Federation, and/or WS-Trust STS endpoints—a pop-up window displays only those endpoints related to the federation protocols enabled in Server Settings (see “Choosing Roles and Protocols” on page 106). These endpoints are built into PingFederate and cannot be changed.

Your federation partners or STS clients need to know the applicable IdP Services endpoints to communicate with your PingFederate server. Configured service endpoints for SAML connections are included in metadata export files (see “Exporting Metadata” on page 57).

The table below describes each endpoint:

**Table 22: PingFederate IdP Endpoints**

<table>
<thead>
<tr>
<th>Service</th>
<th>URL and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Logout Service (SAML 2.0)</td>
<td>/idp/SLO.saml2&lt;br&gt;The URL that receives and processes logout requests and responses.</td>
</tr>
<tr>
<td>Single Sign-on Service (SAML 2.0)</td>
<td>/idp/SSO.saml2&lt;br&gt;The SAML 2.0 implementation URL that receives authentication requests for processing.</td>
</tr>
<tr>
<td>Artifact Resolution Service (SAML 2.0)</td>
<td>/idp/ARS.ssaml2&lt;br&gt;The SOAP endpoint that processes artifacts returned from a federation partner to retrieve the referenced XML message on the back channel. (See “Important” footnote in this table.)</td>
</tr>
<tr>
<td>Attribute Query Service (SAML 2.0)</td>
<td>/idp/attrsvc.ssaml2&lt;br&gt;The SAML implementation that receives and processes attribute requests. (See “Important” footnote in this table.)</td>
</tr>
<tr>
<td>Metadata Service</td>
<td>/&lt;br&gt;The default endpoint (empty path) from which partners can retrieve Auto-Connect metadata (see “Using Auto-Connect” on page 30).</td>
</tr>
<tr>
<td>Single Sign-on Service (SAML 1.x)</td>
<td>/idp/isx.saml1&lt;br&gt;The SAML 1.x implementation of IdP intersite transfer service (ISX) to which clients are redirected for SSO requests.</td>
</tr>
<tr>
<td>Artifact Resolution Service (SAML 1.x)</td>
<td>/idp/soap.ssaml1&lt;br&gt;The SOAP endpoint that processes artifacts returned from a federation partner to retrieve the referenced XML message on the back channel. (See “Important” footnote in this table.)</td>
</tr>
</tbody>
</table>
As an IdP, you manage connection settings to support the exchange of federation-protocol messages (SAML, WS-Federation, or WS-Trust) with an SP or STS client application at your site.

**Note:** If you are configuring a new connection only for WS-Trust STS, follow the sections in this part of the manual up to and including “General Information” on page 255. Then turn to “WS-Trust STS Configuration” on page 437.

These settings include:
Managing SP Connections

- User attributes you expect to send in an SSO assertion (including STS SAML tokens).
- User attributes that may be sent using the Attribute Query profile (if that profile is used).
- The protocol and, for SAML, the profile you will use, including detailed security specifications (the use of digital signatures, signature verification, XML encryption, and SSL). For more information see the “Supported Standards” chapter in Getting Started.

To continue with the configuration, you and your connection partner must have decided this information in advance (see “Federation Planning Checklist” on page 36). Your federation partner must supply some connection settings and other information (see “Configuration Data Exchange” on page 39).

Tip: If you are configuring connections to more than one partner under SAML 2.0 specifications, or if you intend to add partners in the future, consider using Auto-Connect (see “Configuring SP Auto-Connect” on page 332).

If your agreement includes sending assertions containing attribute values from a local data store, then you need to define the data store during this configuration if you have not done so already (see “Managing Data Stores” on page 116).

Accessing Connections

You can create or modify connections directly via the Main Menu. Note that the menu displays the four most-recently modified connections. To view a list of all SP connections, click the Manage All SP link.

Via the Main Menu

From the Main Menu under IdP Configuration, you can configure a new connection, modify an existing connection, or view connections.

Tip: To copy or delete connections or to recover connection drafts, click Manage All SP (see “Via the Manage Connections Screen” on page 249).
To begin configuring a new connection:

- Click **Create New** under SP Connections on the Main Menu.

**Tip:** The fastest way to create a new connection is to click **Manage All SP** and copy the connection with similar settings (see "Via the Manage Connections Screen" on page 249).

To modify a connection:

1. Click the connection name under SP Connections on the Main Menu.
   
   Only the four most recently edited connections are displayed. To see all connections, including drafts, click **Manage All SP**.

2. On the Activation & Summary screen, click the heading for the information you want to change.

3. Make your change and click **Save**.

**Note:** If **Save** is not available, it means your modification requires other changes or you are editing a screen that is part of a series of subtasks. Click **Next** and continue making indicated changes. The **Done** button indicates that further changes in the task are optional. When you have no further changes, click **Done** and then click **Save** on the task summary screen.
Via the Manage Connections Screen

From the Manage Connections screen you can:

- Create a new connection.
- Modify or copy an existing connection.
- Continue working on a connection draft.
- Delete a connection—if it is not active or referenced in other parts of the configuration (In Use).
- Export individual connection configurations.

**Note:** The connection export function results in an XML file that you can modify and import into another PingTrust server acting in the same federation role (IdP or SP) at your site (see “Connection Management Service” on page 554). You can also automate this process (see “Automating Configuration Migration” on page 78).

- Export metadata about a connection to expedite your partner's corresponding configuration (see “Exporting Metadata” on page 57).

On this screen you can also globally override transaction logging levels set for individual connections or restore connection-based logging (see “Runtime Transaction Logging” on page 47).

**Tip:** A check-mark icon next to a Connection Name indicates that the connection has been checked for configuration errors. For more information about connection-validation features associated with this screen, see “Managing Connection Validation” on page 251.

To access the Manage Connections screen:

- Click Manage All SP under SP Connections on the Main Menu.

To begin configuring a new SP connection:

- Click Create Connection on the Manage Connections screen.

To copy a connection:

1. Click Copy under Action for the connection you want to copy.
Chapter 6
Identity Provider SSO Configuration

2. Update the Connection ID (of the partner) and the Connection Name in the General Info screen (see “General Information” on page 255).
3. Make any further changes needed for the new connection.

**Note:** Outbound Provisioning configurations are not copied for a connection (see “Outbound Provisioning for IdPs” on page 34).

**To edit a connection or continue working on a draft:**
- Click the Connection Name link.
  For a draft, you will return to where you left off.

**To export a connection:**
1. Click Export Connection under Action for the connection.
2. Save the XML file on your file system.
   You can change the name of the file, but keep the XML extension.

**Tip:** You can import the connection programmatically or manually into another instance of PingFederate acting in the same role (see “Connection Management Service” on page 554).

**To export connection metadata:**
1. Click Export Metadata under Action for the connection.
   This action takes you to the Export Metadata screen flow, with the connection selection preset (see “Exporting Metadata” on page 57).
2. Complete the steps remaining in the Export Metadata screen flow (starting at Step 4 under “To export connection metadata:” on page 58).

**To delete a connection:**
1. Under Action, click Delete for the connection.
   (To undo the deletion, click Undelete.)

**Note:** The Delete function is not available if the connection is Active or In Use.
2. To confirm the deletion, click Save.

**To sort the list of connections:**
- Click the arrow next to any column heading to sort the list based on that column.

**To filter the list by Virtual ID, Protocol and/or Status:**
- Select a filter criterion from either or both of the drop-down lists.

**Note:** The Virtual ID displays the default virtual server ID of the connection, if any (see “General Information” on page 255).

**To override connection-based transaction logging:**
1. Select On under Logging Mode Override.
2. Choose the logging mode you want to use for all connections.
To restore connection-based transaction logging:
- Select Off under Logging Mode Override.

Managing Connection Validation

By default PingFederate automatically validates all existing connections before displaying the Manage Connections screen. This validation ensures that any updates to supporting components—adapters or data-store configurations, for example—have not invalidated any connection settings.

If such errors are found, a warning icon appears next to the Connection Name.

To correct errors:
- Click the Connection Name to reach the top-level task in which reconfiguration is needed, and to see the error message. Then navigate into deeper tasks using Configure... buttons to find a link to the screen that needs updating.

(For more information about console navigation, see “About Tasks and Steps” in the “Console Navigation” chapter of Getting Started.)

Note that the connection validation time increases with the number of connections and when connections are configured to access data stores for attribute mapping. Consequently, there may be noticeable delays in displaying the Manage Connections screen. For this reason, PingFederate provides a way to turn off the automatic validation under Server Settings (see “Setting System Options” on page 110).

When validation is turned off, administrators can check connections manually on the Manage Connections screen. A question-mark icon indicates the connection has not been validated. In addition, Action links are disabled (except for Delete, if the connection is Inactive and/or In Use). You may, however, still edit the connection by clicking its name.

When automatic validation is disabled, use one of the following procedures to validate connections:

To validate a single connection:
- Click icon next to the Connection Name.

Note: Validating a single connection does not check connectivity for any configured data-store lookups (see “Selecting Assertion Mapping” on page 269). However, this check is performed when you access the connection for editing.

To validate all connections (including for data-store connectivity):
1. Click Check All Connections for Errors.
2. When prompted, click Yes.
Chapter 6
Identity Provider SSO Configuration

Choosing a Connection Template

On the Connection Template screen (shown only for a new connection), you can choose a quick-connection template if your installation includes an optional PingFederate SaaS Connector (see “Outbound Provisioning for IdPs” on page 34).

Tip: When you select a Connection Template, many connection settings are configured for you automatically. For information about using a template, refer to the Quick Connection Guide available with your PingFederate SaaS Connector.

To configure a connection without a template, click Next.

Choosing a Connection Type

If you are not using a connection template (which preconfigures browser-based SSO), indicate on the Connection Type screen whether the connection to this partner is for Browser SSO, WS-Trust STS, and/or Outbound Provisioning (see “Connection Types” on page 5).

Note: You can add STS, OAuth, and Outbound Provisioning support to any existing SSO connection, or vice versa, at any time.
Managing SP Connections

If your federation deployment supports multiple protocols and you are not using a connection template, then for new SSO connections you can also select the applicable protocol on the Connection Type screen (see “Choosing Roles and Protocols” on page 106).

**Note:** If your partner’s deployment also supports multiple protocols and you intend to communicate using more than one, then you must set up a separate connection for each protocol.

- To configure a connection for secure browser-based SSO, select Browser SSO Profiles and the Protocol (if necessary).

- To configure a connection for WS-Trust STS, make that selection and then select a Default Token Type.

  The Default Token Type, either SAML 1.1 or 2.0, is used when a Web Service client does not specify in the token request what token type the STS should issue.

  **Note:** The Default Token Type does not need to match the Protocol indicated on the screen for SSO (when applicable).

- To configure a connection for Outbound Provisioning, make that selection and then select the Outbound Provisioning Type (if multiple outbound provisioning drivers are installed, such as SCIM, Google, or Salesforce).

  **Note:** The Outbound Provisioning option is active only after you enable the Outbound Provisioning protocol on the Roles & Protocols screen (see “Choosing Roles and Protocols” on page 106).

- (Optional) If your PingFederate license manages connections by groups, then you can select a group for this connection.

  This option is not displayed for unrestricted or other types of licenses.
Choosing Connection Options

On the Connection Options screen, you can enable the Browser SSO and Attribute Query options for the current connection. In addition, for browser-based SSO you can enable IdP Discovery for the current connection.

**Note:** This screen is presented only for browser-based SSO connections (see “Choosing a Connection Type” on page 252).

To create a connection for browser-based SSO, select Browser SSO.

To enable IdP Discovery for this connection, select IdP Discovery.

**Note:** The IdP Discovery checkbox is only enabled if IdP Discovery is configured and the Domain Cookie Setting “IdP using a common domain service to advertise the ability to authenticate a principal” is enabled (see “Standard IdP Discovery” on page 132).

To create a connection for attribute query, select Attribute Query. See “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started.

Click Next

Importing Metadata

If you are using one of the SAML protocols (without a connection template) and have received a metadata file from your partner, click Browse on the Import Metadata screen, select the file, and click Next.

**Note:** If the endpoints in the metadata file share the same base URL (protocol, hostname, and port), PingTrust uses this information to populate the Base URL field (see “General Information” on page 255). Consequently, individual endpoints on other screens do not include this information—only relative paths are shown.

**Note:** If you are importing a signed metadata file that does not include the certificate and public key, you will be asked to import the certificate needed to verify the XML signature (see the next section).

If you are not using a metadata file, click Next on the Import Metadata screen.
Importing a Verification Certificate

The Import Certificate screen appears only if the metadata file you have chosen to import is signed and the certificate needed to verify the signature is not contained in the file.

- Click **Browse** to locate and open the signature verification certificate for this partner.

Viewing the Metadata Summary

The Metadata Summary screen provides security information about an imported metadata file, including whether the file was signed and, if so, the trust status of the certificate used to verify the signature.

General Information

On the General Info screen, you provide a required unique identifier and display name for a connection, as well as optional contact information. In addition, on this screen you can set the level of transaction logging for this connection partner (see “Runtime Transaction Logging” on page 47).
### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partner’s Entity ID/Audience/Partner’s Realm (Connection ID)</td>
<td>(Required) The published, protocol-dependent, unique identifier of your partner. For a SAML 2.0 connection, this is your partner’s SAML Entity ID. For a SAML 1.x connection, this is the Audience your partner advertises. For a WS-Federation connection, this is your partner’s Realm. This ID may have been obtained out-of-band or via a metadata file if you are using a SAML protocol (see ). For STS-only connections, this ID can be any unique identifier.</td>
</tr>
<tr>
<td>Connection Name</td>
<td>(Required) A plain-language identifier for the connection—for example, a company or department name. This name is displayed in the connection list on the Main Menu.</td>
</tr>
<tr>
<td>Virtual Server IDs</td>
<td>If you want to identify your server to this connection partner using an ID other than the one you specified under Server Settings (see ), enter a virtual server ID in this field and click Add. Enter additional virtual server IDs as needed (see ).</td>
</tr>
<tr>
<td>Base URL</td>
<td>The fully qualified hostname and port on which your partner’s federation deployment runs (e.g., <a href="https://www.pingidentity.com:9031">https://www.pingidentity.com:9031</a>). This entry is an optional convenience, allowing you to enter relative paths to specific endpoints, instead of full URLs, during the configuration process.</td>
</tr>
<tr>
<td>Company</td>
<td>The name of the partner company to which you are connecting.</td>
</tr>
<tr>
<td>Contact Name</td>
<td>The contact person at the partner company.</td>
</tr>
<tr>
<td>Contact Number</td>
<td>The phone number of the contact person at the partner company.</td>
</tr>
<tr>
<td>Contact Email</td>
<td>The email address for the contact person at the partner company.</td>
</tr>
<tr>
<td>Logging Mode</td>
<td>The level of transaction logging applicable for this connection (see ).</td>
</tr>
</tbody>
</table>

**To reach this screen:**

1. Click a connection name on the Main Menu.
   
   Click **Manage All SP**, if needed, to see a full list of connections.

2. Click **General Info** under the SP Connection tab.
For a new connection:

► Fill in the information needed and click **Next**.

Connection ID and Connection Name are required (see “Field Descriptions” above).

Note that the values in Virtual Server IDs identify your own federation deployment for this connection only and override the ID you specified under Server Settings (see “Federation Server Identification” on page 37).

For an existing connection:

► If you are editing existing information, modify the fields as needed and click **Save**.

### Configuring Browser-Based SSO

Browser-based SSO (also, Browser SSO) is another term for secure SSO, which relies on a user’s Web browser and HTTP to broker XML identity-federation messaging between an IdP and an SP (in contrast to WS-Trust STS messaging, which is typically application-driven across the back channel and does not require browser mediation).

To continue, click **Configure Browser SSO**.

### Configuration Steps

Many steps involved in setting up a federation connection are protocol-independent; that is, they are required steps for all connections, regardless of the associated standards (see the “Supported Standards” chapter in Getting Started). Also, for any given connection, some configuration steps are required under the applicable protocol, while others are optional. Still others are required only based on certain selections. The PingFederate administrative console determines the required and optional steps based on the protocol and dynamically presents additional requirements or options based on selections.

The following sections provide sequential information about every step you might encounter while configuring browser-based SSO, regardless of the protocol you are using for a particular connection.

**Note:** The configuration screens represented in this chapter show “SAML 2.0” in their left corners, unless they are exclusive to WS-Federation or SAML 1.x setup requirements. When the SAML 2.0 screens are also applicable to SAML 1.x and/or WS-Federation connections, the SAML 2.0 representations and discussions also apply to the other protocols, unless otherwise indicated.

After configuring SSO settings, you will normally need to configure authentication credentials, the range of which depends on your SSO selections (see “Configuring
Credentials” on page 306). Also, other configuration tasks may remain to be configured for new or modified connections, depending on selected connection options (see “Choosing Connection Options” on page 254).

**Important:** For new connections you must completely configure these SSO settings and subsequent tasks before you can save the connection on the Activation & Summary screen. Until then, the *configuration is temporary and can be lost*, the console times out after several minutes of inactivity. At any time, however, you can click **Save Draft**, which is available on most screens after you enter General Information (see “Console Buttons” in the “Console Navigation” chapter of *Getting Started*).

Use the lists and links (or page references) below to find specific information about steps that may apply to your SSO connection requirements:

**SAML 2.0 SSO Configuration Steps**
- “Choosing Profiles (SAML 2.0)” on page 259
- “Setting an Assertion Lifetime” on page 261
- “Assertion Creation” on page 261
  - “Choosing an Identity Mapping Method” on page 262
  - “Creating an Attribute Contract” on page 264
  - “IdP Adapter Mapping” on page 265
- “Configuring Protocol Settings” on page 290
  - “Setting Assertion Consumer Service URLs (SAML)” on page 291
  - “Specifying SLO Service URLs (SAML 2.0)” on page 294
  - “Choosing Allowable SAML Bindings (SAML 2.0)” on page 295
  - “Setting an Artifact Lifetime (SAML)” on page 295
  - “Specifying Artifact Resolver Locations (SAML 2.0)” on page 296
  - “Defining Signature Policy” on page 297
  - “Configuring XML Encryption Policy (SAML 2.0)” on page 297

**WS-Federation SSO Configuration Steps**
- “Setting an Assertion Lifetime” on page 261
- “Assertion Creation” on page 261
  - “Choosing an Identity Mapping Method” on page 262
  - “Creating an Attribute Contract” on page 264
  - “IdP Adapter Mapping” on page 265
- “Configuring Protocol Settings” on page 290
  - “Defining a Service URL (WS-Federation)” on page 293

**SAML 1.x SSO Configuration Steps**
- “Setting an Assertion Lifetime” on page 261
- “Assertion Creation” on page 261
  - “Choosing an Identity Mapping Method” on page 262
  - “Creating an Attribute Contract” on page 264
  - “IdP Adapter Mapping” on page 265
- “Configuring Protocol Settings” on page 290
Choosing Profiles (SAML 2.0)

A SAML profile is the message-interchange scenario that you and your federation partner have agreed to use (see “Federation Planning Checklist” on page 36). For SAML 2.0, PingFederate supports all IdP- and SP-initiated SSO and SLO profiles.

For information on typical SSO/SLO profile configurations, including illustrations, see the “Profiles” sections in the “Supported Standards” chapter in Getting Started.

Note: This screen is not presented for SAML 1.x connections because IdP SSO is assumed, the SLO profiles are not supported, and the server supports SP-initiated SSO automatically (see “SAML 1.x Profiles” in the “Supported Standards” chapter of Getting Started).

The screen is also not presented for WS-Federation connections because profile selection is not required (see “WS-Federation” in the “Supported Standards” chapter of Getting Started).

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click SAML Profiles on the Summary screen.

To configure profiles:
1. Select the profile(s) applicable to this connection and click Next.
   - You must select an SSO profile before you can enable SLO.
2. Continue through the remaining connection-configuration tasks.

Configuring IdP-Initiated SSO

When PingFederate is operating as an IdP, the IdP-initiated SSO profile configuration defines the message-transport mechanisms (bindings) your enterprise has agreed to use for this partner, plus optional digital signature requirements for outbound assertions (see “Security Infrastructure” on page 25).

For illustrations of typical profile and binding scenarios, see the “Profiles” sections in the “Supported Standards” chapter in Getting Started.
For this configuration you need to know:

- The transport binding(s) to which you and your partner have agreed
- The certificate to be used for signing assertions (not always required for the artifact binding)
- The URL(s) of your partner’s Assertion Consumer Service(s)

**Configuring SP-Initiated SSO**

The SP-initiated profile configuration for SSO defines the message-transport mechanisms (bindings) and security requirements for receiving authentication requests and sending assertions when your SP partner initiates SSO transactions (see “Single Sign-on” in the “Supported Standards” chapter of Getting Started).

For SAML 1.x, the SP-initiated SSO profile is also known as the “destination-first” profile, which was added as a supported “non-normative” use case after the release of the SAML 2.0 specifications. As an IDP, you do not need to configure this profile; when the SP sends an authentication request to your SSO Service endpoint, PingFederate will handle the response automatically.

For illustrations of typical profile and binding scenarios, see the “Profiles” sections in the “Supported Standards” chapter in Getting Started.

For this configuration you will need to know:

- The endpoint URL(s) for your SP’s Assertion Consumer Service(s)
- The transport bindings that you and your partner have agreed upon inbound and outbound
- The certificates you will use to sign outbound assertions and to verify incoming digital signatures from your SP, when either is required

When Artifact is an allowable inbound SAML binding, you also need to know the endpoint(s) to your partner’s Artifact Resolution Service(s) and the SOAP client authentication mechanism to use: either HTTP Basic, SSL client certificates, a digital signature, or a combination of these mechanisms.

**Configuring IdP-Initiated SLO**

The SAML 2.0 IdP-initiated SLO profile configuration defines the message-transport mechanisms (bindings) and security requirements for exchanging SLO requests and responses.

For more information about SLO, see “Single Logout” in the “Supported Standards” chapter of Getting Started.

For this configuration you need to know:

- The transport bindings to which you and your partner have agreed to send SLO requests and receive responses
- The certificates to be used for signing outgoing messages and for verifying incoming digital signatures from your SP (not always required for the artifact binding)
- The URL(s) of your SP’s Single Logout Service(s)

**Configuring SP-Initiated SLO**

The SAML 2.0 SP-initiated SLO profile configuration defines the message-transport mechanisms (bindings) and security requirements that you and your partner have agreed upon for exchanging SAML requests and responses.
For more information about SLO, see “Single Logout” in the “Supported Standards” chapter of Getting Started.

For this configuration you need to know:

- The transport bindings that you and your partner have agreed upon to send SLO requests and receive responses
- The certificates to be used for signing outgoing messages and for verifying incoming digital signatures from your SP (not always required for the artifact binding)
- The URL(s) of your SP’s Single Logout Service(s)
- The URL of your SP’s Artifact Resolution Service(s)—if this binding and endpoint are different from your SSO configuration—and SOAP client authentication requirements

### Setting an Assertion Lifetime

Identity-federation standards require a window of time during which an assertion is considered valid. Each assertion has a time-stamp XML element as well as elements indicating the allowable lifetime of the assertion (in minutes) before and after the assertion time stamp.

<table>
<thead>
<tr>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
</tr>
<tr>
<td>Minutes Before</td>
</tr>
<tr>
<td>Minutes After</td>
</tr>
</tbody>
</table>

To change the default times:

- (Optional) Edit the desired setting(s) and click Next or Save.

### Assertion Creation

As an IdP, you must specify how PingFederate obtains user-authentication information and use it to create assertions appropriate for your SP partner, including additional user attributes as needed. This configuration includes:

- Choosing an identity-mapping method (see “Choosing an Identity Mapping Method” next).
- Defining the attribute contract you will use with this partner, if any (see “Creating an Attribute Contract” on page 264).
- Configuring instances of one or more adapters (see “IdP Adapter Mapping” on page 265) and specifying how they are used to fulfill the attribute contract.
Chapter 6
Identity Provider SSO Configuration

Choosing an Identity Mapping Method

PingFederate allows your SP partner to use either account linking or account mapping to associate remote users with local accounts for SSO between business partners (see “Identity Mapping” on page 17). At the Identity Mapping step, you choose the type of name identifier your partner needs to facilitate one of these options. You and your partner may want to decide in advance which option to use (see “Federation Planning Checklist” on page 36).

The choices of name-identifier types depend on which protocol you are using:
- For information about SAML selections, see “SAML Name ID Selections” next.
- For information about WS-Federation selections, see “WS-Federation Name ID Selections” on page 263.

SAML Name ID Selections

The allowable types of name identifiers for SAML connections are described below. These choices affect how SPs make use of account mapping or account linking.

If your SP is using account linking, then establishing an attribute contract is not required. Depending on your partner agreement, however, you may choose to supplement the account link with an attribute contract. In this configuration the account link is used to determine the user's identity, while the additional attributes
might be used for authorization decisions, customized Web pages, and so on, at the SP site (see “About Attributes” on page 19).

Important: If you have previously set up a configuration to use an attribute contract and want to change the configuration to use account linking without additional attributes, then the existing attribute contract will be discarded.

Account linking can be used with either a clear, standard name identifier or an opaque pseudonym.

► If you want to send a known attribute to identify a user—for example, a username or email address—then select Standard.

► If you and your partner have agreed to use an opaque persistent name identifier (often used for account linking), then select Pseudonym on the Identity Mapping screen.

The pseudonym is based on values pulled from the IdP adapter instance used to authenticate the user. You select these values when you configure IdP adapter instances (see “Setting Pseudonym Values and Masking” on page 234).

To set up an attribute contract to use in conjunction with an opaque identifier, select the checkbox next to “Include attributes . . .” after selecting Pseudonym.

► Select Transient to enhance the privacy of a user’s identity. Unlike a pseudonym, a transient identifier is different each time a user initiates SSO (see “Account Linking” on page 17).

A typical application for this selection might be, for example, when an SP provides generalized group accounts based on organizational rather than individual identity.

To set up an attribute contract to use in conjunction with an opaque identifier, select the checkbox next to “Include attributes . . .” after selecting Transient.

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click Identity Mapping on the Summary screen.

WS-Federation Name ID Selections

For WS-Federation purposes a name identifier is a uniquely identifying user attribute.

► Make one of the selections described below:
  • Email Address: This attribute is commonly used as a unique identifier for SSO and SLO. Make this selection, for example, if a user logs in using an email address or if the information is available for lookup in a local data store.
  • User Principal Name: The username or other unique ID of the subject initiating the transaction. Make this selection, for example, if a username
will be available from the current user session as part of a cookie or can be
derived from a local data store.

- **Common Name**: This selection provides for anonymous SSO to your SP,
generally using a hard-coded generalized logon. Make this selection if your
partner agreement involves a many-to-one use case—for example, if the SP
has a group account set up for all users in a particular domain.

Later, you will map your choice to the `SAML_SUBJECT` attribute in the SAML
assertion (see “Mapping Default Attribute Contract Fulfillment” on page 289).

### Creating an Attribute Contract

An attribute contract is the set of user attributes that you and your partner have
agreed will be sent in SAML assertions for this connection (see “Attribute
Contracts” on page 19). You identify these attributes on this screen.

If you are sending a "standard" name identifier (see “Choosing an Identity Mapping
Method” on page 262), then the contract includes the default `SAML_SUBJECT`,
which identifies the user in the assertion. You will configure this variable later to
contain a user ID or another agreed-upon attribute—for instance, an email address—
that uniquely identifies the user (see “Attribute Contract Fulfillment” on page 282).

**Note:** Creating an attribute contract is optional if you are sending either
a pseudonym or a transient identifier to your connection partner (see
“Choosing an Identity Mapping Method” on page 262).

---

**To reach this screen:**

1. Click the connection name on the Main Menu.
   - Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the **SP Connection** tab.
3. Click **Configure Browser SSO**.
4. Click **Assertion Creation** under the **Browser SSO** tab.
5. Click **Configure Assertion Creation**.
6. Click **Attribute Contract** on the Summary screen.
   - If this step is not in the list, then you have chosen to send either a pseudonym
     or a transient identifier without additional attributes (see “Choosing an Identity
     Mapping Method” on page 262).

**Note:** For WS-Federation connections, the Subject Name Format is
determined by the previous screen (see “WS-Federation Name ID
Selections” on page 263).
To add an attribute:

1. Enter the attribute name in the text box.
   
   Attribute names are case-sensitive and must correspond to the attribute names expected by your partner.

   **Tip:** You can add a special attribute, SAML_AUTHN_CTX, to indicate to the SP (if required) the type of credentials used to authenticate to the IdP application—authentication context. Map a value for the authentication context on the attribute-mapping screen later in the configuration, from any available attribute source (see “Attribute Contract Fulfillment” on page 282).

2. (Optional) Select the Attribute Name Format.

   **Note:** This option is not available for SAML 1.0 connections.

   Change the default Name Format selection if you and your SP partner have agreed to a specific format (see “Name Formats” on page 20).

   **Note:** If needed, an administrator can customize name-format alternatives via the custom-name-formats.xml configuration file located in this directory:

   <pf_install>/pingfederate/server/default/data/config-store

3. Click Add.

   **(Optional) For SAML connections, to modify the Subject Name Format:**

   ▶ Make your selection in the drop-down list and click Done.

   The default selection is usually applicable, unless you and your partner have agreed to a different format specification (see “Name Formats” on page 20).

   **Note:** This selection is not available if you are sending a pseudonym as the subject, or a transient identifier (see “Choosing an Identity Mapping Method” on page 262).

To modify an attribute name or format:

1. Click Edit under Action for the attribute.

2. Make the change and click Update.

   **Note:** If you change your mind, ensure that you click the Cancel link in the Actions column, not the Cancel button, which discards any other changes you might have made in the configuration steps.

To delete an attribute:

▶ Click Delete under Action for the attribute.

**IdP Adapter Mapping**

IdP adapters are responsible for handling user authentication as part of an SSO operation (see “SSO Integration Kits and Adapters” on page 14). A configured and deployed adapter in PingFederate is known as an adapter instance. The same instance may be mapped by multiple connections.
Map one or more IdP adapter instances into each SP connection so that when a user authenticates with a particular external identity management system the user attributes are returned to PingFederate.

When adapters are restricted to certain virtual server IDs, the allowed IDs are displayed under the Virtual Server IDs column.

Regardless of how many IdP adapter instances are mapped in an SP connection, PingFederate uses only one instance to authenticate a user. Because each instance may return different user attributes, each IdP adapter mapping must define how the attribute contract is fulfilled; you must map attributes from each adapter—and/or attributes retrieved from your local data stores—into the assertions PingFederate sends to this SP to fulfill the attribute contract.

You begin this configuration on the IdP Adapter Mapping screen, where you choose to map instances of IdP adapters. If you have not yet configured an instance of the adapter you intend to use within this SP connection, see “Configuring IdP Adapters” on page 230.

To modify an existing Adapter Instance:
► Click its Name link.

To begin configuring an Adapter Instance for this connection:
► Click Map New Adapter Instance.

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.

Selecting an Adapter Instance

A configured and deployed adapter in PingFederate is known as an adapter instance. The same adapter instance may be mapped by multiple connections (see “Configuring IdP Adapters” on page 230).

You can use attributes returned from the adapter (the adapter contract) to fulfill the attribute contract with this partner, and/or use them to look up additional attributes.
in a user-data store. You make this choice on the Assertion Mapping screen (see “Selecting Assertion Mapping” on page 269).

![Managing SP Connections](image)

Choose an Adapter Instance from the drop-down list and click Next. (Optional) To override any IdP adapter instance, select the Override Instance Settings checkbox (see “Overriding IdP Adapter Instances” on page 267).

To create or change an adapter instance, as needed, click Manage Adapter Instances.

To reach this screen:

1. Click the connection name on the Main Menu.
   
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Adapter Instance on the Summary screen.

Overriding IdP Adapter Instances

Overrides at the connection level simplify adapter management by allowing you to create a small set of adapter instances that define the base configuration requirements and then overriding settings at the connection level as needed.

You may override any IdP adapter settings at the connection level either during or after connection mapping.

Alternatively, you can override any adapter instance to apply to several connections, see “Hierarchical Plug-in Configurations” on page 17 for more information about adapter overrides.

**Note:** Any changes to the base adapter instance are propagated to a connection provided the same changes are not overridden for the connection.
Chapter 6
Identity Provider SSO Configuration

Click **Override Instance Settings**.

On each of the settings screens, make your changes, and then click **Next**. When you are finished, click **Done** to continue with IdP adapter mapping.

**Note:** Override adapter-setting screens are functionally identical to those used for creating a new adapter connection. Refer to the table below to find sections in this manual containing configuration information and procedures.

The display of some screens listed in the table depends on the type of adapter you are configuring.

For information about the override adapter-setting screens, depending on the type of setting, use the following table:

<table>
<thead>
<tr>
<th>Override Screen</th>
<th>Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter</td>
<td>See “Configuring an IdP Adapter” on page 232.</td>
</tr>
<tr>
<td>Actions</td>
<td>See “Invoking Adapter Actions” on page 233.</td>
</tr>
<tr>
<td>Adapter Attributes</td>
<td>See “Setting Pseudonym Values and Masking” on page 234.</td>
</tr>
</tbody>
</table>

To remove any adapter connection override, clear the **Override Instance Settings** checkbox, and then complete the rest of the setup.

**To reach this screen:**
1. Click the connection name on the Main Menu.
2. Click **Manage All SP**, if needed, to see a full list of connections.
3. Click **Browser SSO** under the SP Connection tab.
4. Click **Configure Browser SSO**.
5. Click **Assertion Creation** under the Browser SSO tab.
6. Click **Configure Assertion Creation**.
7. Click **IdP Adapter Mapping** on the Summary screen.
8. Click **Adapter Instance** on the Summary screen.
9. Select the **Override Instance Settings** checkbox.
10. Click Next.

**Restricting an IdP Adapter to certain Virtual Server IDs**

When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), you have the option to enforce authentication requirements by restricting an adapter to certain virtual server IDs. This optional setting can be applied to each adapter added to the connection using virtual server IDs. By default, no restriction is imposed.

To restrict an adapter to a subset of the available virtual server IDs:

1. Select the Restrict Virtual Server IDs checkbox.
2. In the Allowed Virtual Server IDs area, select one or more virtual server IDs that you want to allow for this IdP adapter.
3. Click Next.

To reach this screen:

1. Click the connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Virtual Server IDs on the Summary screen.

**Note:** The Virtual Server IDs screen is only available for connections using at least one virtual server ID, see “Federation Server Identification” on page 37.

**Selecting Assertion Mapping**

For SAML assertions, you can query local user-data stores to help fulfill the attribute contract, in conjunction with attribute values supplied by the IdP adapter you are using with PingFederate (see “SSO Integration Kits and Adapters” on page 14).

When using data stores, you can use one or more data stores to look up attributes for a single mapping. Alternatively, you can define alternate data stores and a failsafe mapping.

The attributes supplied by the adapter you have chosen are shown under Adapter Contract on the Assertion Mapping screen.
Chapter 6
Identity Provider SSO Configuration

If you use only the Adapter Contract values, then you map values for the attribute contract next (see “Mapping Default Attribute Contract Fulfillment” on page 289).

If you choose to retrieve additional attributes, then you identify data stores and specify lookup queries next (see “Configuring Attribute Sources and User Lookup” on page 270).

To reach this screen:
1. Click the connection name on the Main Menu. Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Assertion Mapping on the Summary screen.

Tip: To determine whether you need to look up additional values, compare the attribute contract against the adapter contract (see “Creating an Attribute Contract” on page 264 and “IdP Adapter Mapping” on page 265). If the attribute contract requires more information, determine whether local data stores can supply it. (You can also choose to use text constants or expressions for certain information—see “Mapping Default Attribute Contract Fulfillment” on page 289.)

Configuring Attribute Sources and User Lookup

Attribute sources are specific data store or directory locations containing information that may be needed for the attribute contract (see “Creating an Attribute Contract” on page 264). Attribute sources can be reused across connections to other SP partners.

You can use more than one attribute source when mapping values to an assertion:

• Set up search parameters for your data stores, including “fall-through” searches. If any search fails, the next search executes until all searches are exhausted, at
which point a configured failsafe mapping executes. If the failsafe mapping is not configured, the SSO transaction fails.

**Note:** Queries are executed in the order of Attribute Sources shown. Use the up/down arrows as needed to adjust the order. Note, however, that data can originate from only one source.

- Configure separate data stores to look up attributes for a single mapping. If any search fails to find a user, then the SSO transaction fails.

To configure an attribute source:

- Click **Add Attribute Source** and complete the setup steps (see “Attribute Source Setup” next).

**To modify an attribute source configuration:**

1. Click the attribute source Description link.
2. Click **Save** on the screen you change.

**Note:** Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click **Save** or **Done** when either of those options appears.

**To reach this screen:**

1. Click the connection name on the Main Menu.
   
   Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the SP Connection tab.
3. Click **Configure Browser SSO**.
4. Click ** Assertion Creation** under the Browser SSO tab.
5. Click **Configure Assertion Creation**.
6. Click **IdP Adapter Mapping** on the Summary screen.
7. Click the Adapter Instance Name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping tab.

If this step is not listed, then this instance is configured to use adapter values only (see “Selecting Assertion Mapping” on page 269).
### Attribute Source Setup

For attribute-source setup information, refer to the sections indicated in the following steps.

**Note:** As you make selections on configuration screens, ensure that you allow enough time for PingFederate to access your data store and populate drop-down lists.

1. See “Selecting a Data Store” (next section).
2. See the following sections in this manual, depending on the type of data store:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Related Manual Section</th>
</tr>
</thead>
</table>
| JDBC            | • “Selecting a JDBC Database Table and Columns” on page 273  
                 | • “Configuring a Database Filter (WHERE Clause)” on page 275 |
| LDAP            | • “Configuring an LDAP Directory Search” on page 277  
                 | • “Configuring an LDAP Filter” on page 280 |
| Custom          | • “Configuring Custom Source Filters” on page 282  
                 | • “Selecting Custom Source Fields” on page 282 |


### Selecting a Data Store

This screen allows you to choose a data store from a previously configured list (see “Managing Data Stores” on page 116). Attribute values extracted from this data store are used to help fulfill the attribute contract for this partner (see “Creating an Attribute Contract” on page 264).

**To reach this screen:**

1. Click the connection name on the Main Menu.
   - Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the SP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **Assertion Creation** under the Browser SSO tab.
Managing SP Connections

5. Click **Configure Assertion Creation**.
6. Click **IdP Adapter Mapping** on the Summary screen.
7. Click the Adapter Instance Name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping tab.
   
   If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).
9. Click the attribute source Description link.

**To define an attribute source:**

1. Enter an Attribute Source Id to uniquely identify the data source for the mapping.

   **Note:** This field appears only if you are retrieving additional attributes from multiple data stores using one mapping (see “Selecting Assertion Mapping” on page 269).

2. Use Attribute Source Description to specify an attribute source name that distinguishes this user lookup for the selected data store.

   **Note:** When using multiple data stores for one mapping, PingFederate appends this description to the data store type in the Source list on the Attribute Contract Fulfillment screen (see “Attribute Contract Fulfillment” on page 282).

3. Choose an Active Data Store and click **Next**.

   A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click **Manage Data Stores** to add it (see “Managing Data Stores” on page 116).

**Selecting a JDBC Database Table and Columns**

When you choose to use a database source for attributes, you follow this path through the configuration steps.

On this screen you begin to specify exactly where additional data can be found to complete the attribute contract when you send an assertion to this SP (see “Creating
Important: (For MySQL users) To allow for table and column names that may contain spaces, PingFederate inserts double quotes around the names at runtime. To avoid SQL syntax errors resulting from the quotes, add the session variable `sql_mode=ANSI_QUOTES` to the connection string. For example:

```
```

For more information, see:

- [dev.mysql.com/doc/refman/5.0/en/identifiers.html](http://dev.mysql.com/doc/refman/5.0/en/identifiers.html)

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Lists the table structure that stores information within a database. Some databases, such as Oracle, require selection of a specific schema for a JDBC query. Other databases, such as MySQL, do not require selection of a schema.</td>
</tr>
<tr>
<td>Table</td>
<td>The name of the table contained in the database. Use the drop-down to change the table.</td>
</tr>
<tr>
<td>Columns to return from SELECT</td>
<td>Displays selected table columns. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.</td>
</tr>
</tbody>
</table>
To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Attribute Sources & User Lookup under the IdP Adapter Mapping tab.
   If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).
9. Click the attribute source Description link.
10. Click Database Table and Columns.

To select a database table and columns for queries:
1. Choose a Schema file (when applicable) from the drop-down list.
2. Choose a Table from the drop-down list.
3. Choose a name under Columns to Return from Select and click Add Attribute.

Tip: Click Refresh if you are updating an existing configuration and changes may have been made to the database.

Repeat this step for other columns as needed.

Note: You do not need to add a column here for it to be used as part of a search filter (see “Configuring a Database Filter (WHERE Clause)” next). Add only attributes from which you need actual values to pass in an assertion.

Tip: To determine what attributes to look up during a query, click the View Attribute Contract link to see what information must be collected (see “Creating an Attribute Contract” on page 264). Then determine what information is coming in from the session lookup adapter (see “Selecting Assertion Mapping” on page 269). Information not contained in the adapter contract may be pulled from the data store look-up query.

Configuring a Database Filter (WHERE Clause)
The JDBC WHERE clause in PingFederate queries the data table you selected to retrieve a record associated with a particular value (or values) from the assertion. The clause is in the form:
WHERE column1=value1 [AND column2=value2] [OR...]

The left side of the first variable pair uses a column name in the database table you selected (see “Selecting a JDBC Database Table and Columns” on page 273).
The right side generally uses values passed in from your session lookup adapter (variables, including the correct formatting, are listed under Adapter Values—see “Configuring IdP Adapters” on page 230).

**Note:** If you are retrieving attributes from multiple data stores using one mapping, attributes available from other sources, if previously configured, are listed near the bottom of the screen. For more information on multiple data-store mapping, see “Multiple Data Source Attribute Mapping” on page 22.

You can also apply additional search criteria from your own database, using any other columns from the targeted table.

**Tip:** Click "View List of Columns . . ." to see a list from which to copy and paste.

For more information about WHERE clauses, consult your DBMS documentation.

**EXAMPLE:**

```
userid='${username}'
```

In this example `userid` is the name of a column in the JDBC data store. On the right side, `${username}` returns the value of the `username` variable from the IdP adapter.

**Important:** You must use the `$()` syntax to retrieve the value of the enclosed variable and use single quotation marks around the `$()` characters.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where</td>
<td>WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: WHERE email='<a href="mailto:clive@company.com">clive@company.com</a>'.</td>
</tr>
</tbody>
</table>

To reach this screen:

1. Click the connection name on the Main Menu.
   * Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click Attribute Sources & User Lookup under the IdP Adapter Mapping tab.
   * If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).
7. Click the attribute source Description link.
8. Click Database Filter from the steps list.

To construct the WHERE clause:

1. Enter the statement in the space provided, following the guidelines and example above.
   * The initial WHERE is optional.
2. Ensure the syntax and variable names are correct.
   * When you click Next, you will map attribute values returned from the database into the assertion (see “Attribute Contract Fulfillment” on page 282).

Configuring an LDAP Directory Search

When you choose to use an LDAP source for attributes, you follow this path through the configuration steps.

On this screen you specify the branch of your LDAP hierarchy where you want PingFederate to look up user data.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. This field is optional if records are located at the LDAP root.</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Determines the node depth of the query. Select Subtree, One level or Object.</td>
</tr>
<tr>
<td>Root Object Class</td>
<td>The class containing the attributes you want.</td>
</tr>
<tr>
<td>Attributes to return from search</td>
<td>A list of attributes added from the drop-down list below. Subject DN is a default attribute, which may be used as the primary user identifier.</td>
</tr>
</tbody>
</table>

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping tab.
   
   If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).

9. Click the attribute source Description link.

10. Click **LDAP Directory Search** from the steps list or fill out the appropriate screens and advance to this screen.

    If you have not yet defined an LDAP data store, see “Selecting a Data Store” on page 272.

**To select LDAP attributes:**

1. (Optional) Enter a **Base DN**.
2. Select a **Search Scope**.
3. Select a **Root Object Class**.
4. Under Attributes to return from search, choose an attribute and click **Add Attribute**.
   
   Note that the attribute **Subject DN** is always returned by default.
5. Repeat the last step for other attributes as needed.
6. (Optional) Change the **Search Scope** or the **Root Object Class** if you want attributes from other locations.

**Note:** You do not need to add an attribute here for it to be used in a search filter (see “Configuring an LDAP Filter”). Add only attributes from which you need actual values to pass in an assertion.

---

**Specifying Encoding for Binary Directory Attributes**

The LDAP Binary Attribute Encoding Types screen appears when a binary attribute is added on the LDAP Directory Search screen. Because binary attribute data cannot be used in an assertion, use this screen to specify which encoding type (Base64 or Hex) you want to apply during attribute contract fulfillment.

For example, Microsoft® Office 365 uses objectGUID, an immutable Active Directory binary attribute associated with user accounts. Office 365 requires this binary data to be Base64-encoded to correlate provisioned federated user data to Active Directory accounts.

**Note:** Attributes are specified as binary when configuring an LDAP connection (see “Specifying LDAP Binary Attributes” on page 126).

---

**To select an attribute encoding type:**

- Select the type of attribute encoding you want to apply for each attribute and click **Next**.
To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Attribute Sources & User Lookup under the IdP Adapter Mapping tab.
   If this step is not shown, you have elected not to look up attributes in data stores
   (see “Selecting Assertion Mapping” on page 269).
9. Click the attribute source Description link.
10. Click LDAP Binary Attribute Encoding Types from the steps list.

Configuring an LDAP Filter

The LDAP filter queries the data you selected to retrieve a record associated with a
particular value (or values) from the user's session. The filter is in the form:
(attribute=${value}).

The left-side variable is an attribute you selected earlier (see “Configuring an LDAP
Directory Search” on page 277).

The right side generally uses values passed in from your session lookup adapter
(variables, including the correct syntax, are listed under Adapter Values—see
“Configuring IdP Adapters” on page 230).

Note: If you are retrieving attributes from multiple data stores using
one mapping, attributes available from other sources, if previously
configured, are listed near the bottom of the screen. For more
information on multiple data-store mapping, see “Multiple Data Source
Attribute Mapping” on page 22.

You can also apply additional search criteria from your data store, using any other
attributes from the targeted object classes.

Tip: Click “View List of Available LDAP Attributes” for a list from
which you can copy and paste.

For general information about search filters, consult your LDAP documentation.
To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>Narrows a search to locate requested data by either including or excluding specific records. An LDAP filter includes the attributes in the search and the value or range of values that the search is attempting to match. Searches are conducted by using three components: 1) at least one attribute (attribute data type) to search on, 2) a search filter operator that will determine what to match, and 3) the value of the attribute being sought. Searches must have at least one of each of these three components.</td>
</tr>
</tbody>
</table>
Chapter 6
Identity Provider SSO Configuration

8. Click **AttributeSources & UserLookup** under the IdP Adapter Mapping tab.

   If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).

9. Click the attribute source Description link.

10. Click **LDAP Filter** from the steps list.

    If you have not yet defined an LDAP data store, see “Selecting a Data Store” on page 272.

**To construct the LDAP filter:**

1. Enter the statement in the space provided, following the guidelines and example above.

   **Note:** If you used an anonymous binding to create this LDAP connection, your access might be restricted (see “Configuring an LDAP Connection” on page 121).

2. Ensure the syntax and variable names are correct.

3. Click **Next**.

**Configuring Custom Source Filters**

When you choose to use a custom source for attributes, you follow this path through the configuration steps.

On this screen you specify a filter, or lookup query, for your custom data source. This screen display and the syntax of the filter depends on your developer’s implementation of the custom source SDK.

**Selecting Custom Source Fields**

On the Configure Custom Source Fields screen, you can choose from among the fields shown to map to the adapter contract. These choices are supplied by the driver implementation. Select only those needed to fulfill the attribute contract for this partner connection.

**Attribute Contract Fulfillment**

The last step in configuring an attribute source is to map values into the attribute contract (see “Creating an Attribute Contract” on page 264). These are the values included in assertions sent to this SP (provided the information is found in this attribute source).

You map attributes on the Attribute Contract Fulfillment screen.
Map each attribute to fulfill the Attribute Contract from one of these Sources:

- **Adapter**
  Values are returned from the session. When you make this selection, the associated Value drop-down list is populated by the session-lookup adapter (see “SSO Integration Kits and Adapters” on page 14).
  For example, you might choose the adapter attribute `username` to map to `SAML_SUBJECT`.

- **Context**
  Values are returned from the context of the transaction at runtime.

- **LDAP/JDBC/Custom**
  Values are returned from your attribute source. When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes you identified for this Attribute Source (see “Configuring an LDAP Directory Search” on page 277, “Selecting a JDBC Database Table and Columns” on page 273, or “Configuring Custom Source Filters” on page 282).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

**Tip:** You can use an expression to insert an OAuth access token into the assertion for SP-partner use in OAuth transactions (see “About OAuth” on page 10). For information about this feature and a sample expression, refer to the PingFederate SDK Javadoc entry for the class

```java
com.pingidentity.sdk.oauth20.AccessTokenIssuer
```

Javadocs are located in the PingFederate installation:

```html
<pf_install>/pingfederate/sdk/doc/index.html
```
Chapter 6
Identity Provider SSO Configuration

- Text

The value is what you enter. This can be text only, or you can mix text with references to any of the values from the adapter, using the \$\{attribute\} syntax.

You can also enter values from your data store, when applicable, using this syntax:

$\{ds.attr-source-id.attribute\}

where attr-source-id is the Attribute Source Id value (see “Selecting a Data Store” on page 272) and attribute is any of the data store attributes you select.

When using alternate data stores and/or a failsafe mapping, the Attribute Source Id field is not presented. Use the following syntax in this instance:

$\{ds.attribute\}

where attribute is any of the data store attributes you select.

Tip: Two other variables are also available: \$\{SAML_SUBJECT\} and \$\{TargetResource\}. SAML_SUBJECT is the initiating user (or other entity). TargetResource is a reference to the protected application or other resource for which the user requested SSO access; this variable is available only if specified as a query parameter for the relevant PingFederate endpoint (either as \$\{TargetResource\} for SAML 2.0 or \$\{TARGET\} for SAML 1.x—see “Application Endpoints” on page 529).

There are a variety of reasons why you might hard code a text value. For example, if your SP’s Web application provides a service based on your company’s name, you might provide that attribute value as a constant.

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Assertion Creation under the Browser SSO tab.
5. Click Configure Assertion Creation.
6. Click IdP Adapter Mapping on the Summary screen.
7. Click the Adapter Instance Name.
8. Click Attribute Sources & User Lookup under the IdP Adapter Mapping tab.
   If this step is not shown, you have elected not to look up attributes in data stores (see “Selecting Assertion Mapping” on page 269).
9. Click the attribute source Description link.
10. Click Attribute Contract Fulfillment from the steps list.
    (If you have not yet defined a data store, see “Selecting a Data Store” on page 272).
To map attributes:

1. Choose a Source for each Target attribute.
   See “Map each attribute to fulfill the Attribute Contract from one of these Sources:” above.

2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.

3. Click Next.

Specifying Issuance Criteria (Optional)

Use this screen to define criteria PingFederate can evaluate to determine whether to issue an assertion for a user (see "About Token Authorization" on page 24). This token authorization can be used to restrict who can SSO to protected resources.

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   - Adapter – Select to access attributes from the IdP Adapter.
   - Context – Select to use values returned from the context of the transaction at runtime.
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.

Note: This screen does not appear if you chose to use data stores with the failsafe mapping option (see “Overriding IdP Adapter Instances” on page 267).

Note: The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

Note: PingFederate appends a description in parentheses for data stores of the same type (see “Selecting a Data Store” on page 272).
Chapter 6
Identity Provider SSO Configuration

- Mapped Attributes – Select to access the required attributes from the adapter.

2. Select an attribute name.

3. Select the Condition you want to apply.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

Error results are handled differently for SP-initiated and IdP-initiated SSO:

**SP-initiated SSO**
- **SAML** – The Error Result field is used by the StatusMessage element in the response to the SP.
- **Template (WS-Federation)** – The Error Result field is used by the variable $errorDetail in the sourceid-wsfed-idp-exception-template.html template (for more information on this and other user-facing templates, see “Customizing User-Facing Screens” on page 87).

**IdP-initiated SSO**
- **Redirect** – When an InErrorResource URL is provided, the value of the Error Result field is used by an ErrorDetail query parameter in the redirect URL.
- **Template** – When an InErrorResource URL is not provided, the value of the Error Result field is used by the variable $errorDetail in the idp.sso.error.page.template.html template (for more information on this and other user-facing templates, see “Customizing User-Facing Screens” on page 87).

**Note:** Using an error code in the Error Result field allows the error template or a Web application to process the error result in a variety of ways—for example, a redirect to another page or e-mail to an administrator.

If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click **Add**.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.
Managing SP Connections

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

   Note: Expressions must be enabled for the **Show Advanced Criteria** button to appear (see “Enabling and Disabling Expressions” on page 573).

   **Important:** When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

   • Use the in-line editor box to enter the OGNL expression.
     For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
   • Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

   Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

   • Click **Add**. Continue to add expressions, as needed.
   • Click the **Test** link to input values and test the resulting output for the expression.

   Note: For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**

► Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**

► Click **Delete** under Actions for the criteria and then click **Save**.

**Using the Attribute Source Summary Screen**

When you have finished configuring Attribute Sources and User Lookup, you can review the configuration on the Summary screen.

If you need to make any changes, click the heading over the information you want to edit. When you are finished, click **Done** to continue with IdP Adapter Mapping configuration. If you are editing an existing connection, click **Done** on successive screens until you reach the Assertion Creation screen and then click **Save**.

**Specifying a Failsafe Attribute Source**

When a data store is configured and the attribute mappings under Attribute Sources & User Lookup fail to complete the attribute contract, you can choose to
configure a set of “failsafe” Attribute Contract Fulfillment mappings under IdP Adapter Mapping. (For example, you might configure a set of attributes to identify the SSO subject as a “guest” user at the SP.) The failsafe mapping is used instead of the mapping configured with the data-store setup.

**Note:** This screen does not appear if you chose to retrieve attributes from multiple data stores using a single mapping (see “Selecting Assertion Mapping” on page 269).

**Important:** The attribute contract is fulfilled using either the mapping configured under Attribute Sources & User Lookup or the failsafe mapping, not both. In other words, you cannot use the failsafe mapping to fill in missing attributes when some are found via the data-store mapping setup but others are not.

The failsafe mapping is used only when all of the mappings configured in the data-store setup fail to return values for any reason. If any mapping succeeds (an attribute mapped to text, for example), failover does not occur.

Alternatively, you can have PingFederate stop the SSO transaction. This choice depends on your agreement with the SP.

**Note:** If you chose to have PingFederate stop the SSO transaction, the Attribute Contract Fulfillment screen is not presented.

To specify whether to use a failsafe attribute source:

- Make the relevant selection to use either a default set of attributes or to terminate the SSO, and then click **Next**.

To reach this screen:

1. Click the connection name on the Main Menu.
   - Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the SP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **Assertion Creation** under the Browser SSO tab.
5. Click **Configure Assertion Creation**.
6. Click **IdP Adapter Mapping** on the Summary screen.
7. Click the Adapter Instance Name.
8. Click **Failsafe Attribute Source** on the Summary screen.

   This step appears only if you chose to use alternate data stores and/or a failsafe mapping when retrieving attributes (see “Selecting Assertion Mapping” on page 269).

### Mapping Default Attribute Contract Fulfillment

Fulfillment of the attribute contract must be specified whether or not data sources are used. You accomplish this on the Attribute Contract Fulfillment screen, either by choosing to configure default mappings after setting up attribute sources (see “Attribute Contract Fulfillment” on page 282) or if you choose not to set up attribute sources (see “Selecting Assertion Mapping” on page 269).

**Map each attribute to fulfill the Attribute Contract from one of these Sources:**

- **Adapter**
  
  Values are returned from the session. When you make this selection, the associated **Value** drop-down list is populated by the session-lookup adapter (see “SSO Integration Kits and Adapters” on page 14).

  For example, you might choose the adapter attribute **username** to map to **SAML_SUBJECT**.

- **Context**
  
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **Expression** (when enabled)

  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

  **Tip:** You can use an expression to insert an OAuth access token into the assertion for SP-partner use in OAuth transactions (see “About OAuth” on page 10). For information about this feature and a sample expression, refer to the PingFederate SDK Javadoc entry for the class `com.pingidentity.sdk.oauth20.AccessTokenIssuer`. Javadocs are located in the PingFederate installation:


- **Text**

  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the adapter, using the `$(attribute)` syntax.
Chapter 6
Identity Provider SSO Configuration

There are a variety of reasons that you might hard code a text value. For example, if your SP’s Web application provides a consumer service, you might want to supply a particular promotion code.

To map attributes:

1. Choose a Source for each Target attribute (see descriptions of each Source under “Map each attribute to fulfill the Attribute Contract from one of these Sources:” above).
2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.
3. Click Next.

To reach this screen:

1. Click the connection name on the Main Menu.
2. Click Manage All SP, if needed, to see a full list of connections.
3. Click Browser SSO under the SP Connection tab.
4. Click Configure Browser SSO.
5. Click Assertion Creation under the Browser SSO tab.
6. Click Configure Assertion Creation.
7. Click IdP Adapter Mapping on the Summary screen.
8. Click Attribute Contract Fulfillment on the Summary screen.

If you are using data stores for attribute mapping and this step does not appear, see “Specifying a Failsafe Attribute Source” on page 287.

Configuring Protocol Settings

The Protocol Settings screen provides the launching point for configuring bindings, partner endpoints, and other settings needed for the selected SAML profiles (if you
are using SAML 2.0—see “Choosing Profiles (SAML 2.0)” on page 259. The screen also displays configured information.

(For WS-Federation, the configuration of bindings is not applicable.)

To configure Protocol Settings, you need to know:

- For SSO profiles, the URL(s) of your SP’s Assertion Consumer Service(s)
- For SLO profiles, the URL(s) of your SP’s Single Logout Service(s)
- When artifact is an allowable inbound binding, the URL of your SP’s Artifact Resolution Service(s)
- The transport configurations (bindings) that you will use to send and receive data for SSO/SLO connections
- Digital signature policies and certification requirements to which you and your connection partner have agreed
- XML encryption policies to which you and your connection partner have agreed

**Important:** After modifying Protocol Settings, you must click **Done** on the Protocol Settings screen and then **Save** on the Browser SSO screen.

### Setting Assertion Consumer Service URLs (SAML)

At this step for SAML connections, you associate bindings to the Assertion Consumer Service (ACS) endpoint(s) where your SP will receive assertions. This configuration applies to either SSO Profile (see “Choosing Profiles (SAML 2.0)” on page 259).

**Note:** The SP may request that the SAML assertion be sent to one of several URLs, via different bindings. PingFederate uses the defined URL entries on this page to validate the authentication request. However, per SAML specifications, if the request is signed, PingFederate can verify the signature instead; the ACS URL does not necessarily need to be listed here. This is useful for scenarios where an ACS URL might be dynamically generated.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default (SAML 2.0)</td>
<td>A check in this checkbox indicates that the URL configuration in that row will be used as a default.</td>
</tr>
<tr>
<td>Index (SAML 2.0)</td>
<td>Uniquely identifies multiple ACS endpoints.</td>
</tr>
<tr>
<td>Binding</td>
<td>The method of transmission: POST or Artifact.</td>
</tr>
<tr>
<td>Endpoint URL</td>
<td>A location to which the assertion is sent, according to partner requirements.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click Assertion Consumer Service URL on the Summary screen.

To define an Endpoint URL:
1. Select the Binding your partner specifies for the Endpoint.
2. (Optional) Enter an Index value — 0 or 1, for example.
   - For SAML 2.0 the specifications provide for the use of index numbers to identify multiple ACS endpoints. PingFederate supplies this number automatically; however, you can manually set the number to match your partner's configuration as needed.
3. Enter the fully qualified Endpoint URL or just a relative path if you have defined a base URL (see “General Information” on page 255).
4. For SAML 2.0 connections, if this is the default (or only) endpoint, select the checkbox under Default.
5. Click Add.
Setting a Default Target URL (SAML 1.x)

This URL is used whenever PingFederate receives an SSO request from a local application that does not include the user's target resource URL at the SP site. The URL is required regardless of whether you expect your local application(s) to specify the target—to ensure that the server functions correctly during SSO events.

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Target URL</td>
<td>The URL of the target SP resource.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click a connection name on the Main Menu.  
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click Default Target URL on the Summary screen.

Defining a Service URL (WS-Federation)

The Service URL is the WS-Federation endpoint of your SP partner where you send SAML assertions and SLO cleanup messages. The assertions are transmitted within an RSTR (Request for Security Token Response) message in response to a request for authentication from the SP. SLO cleanup messages are sent to WS-Federation SP partners when the IdP receives a user's SLO request. Such cleanup messages indicate that the user's local session has been terminated.

- Enter the fully qualified URL or just the relative path if you have defined a base URL (see “General Information” on page 255).
  You must include the initial slash if you are entering only a relative path.
Chapter 6
Identity Provider SSO Configuration

Specifying SLO Service URLs (SAML 2.0)

At this step you associate bindings to the endpoints where your SP receives logout requests when SLO is initiated at your site and where you send SLO responses when you receive SLO requests from the SP.

This step applies only to SAML 2.0 connections when you select either SLO profile (see “Configuring IdP-Initiated SLO” on page 260 or “Configuring SP-Initiated SLO” on page 260).

Field Descriptions

To add a URL
1. Select the Binding type.
2. Enter the fully qualified URL (or the relative path, if you have specified a base URL—see “General Information” on page 255).
3. (Optional) Enter the Response URL.
4. Click Add.

To edit an endpoint:
1. Click Edit under Action for the endpoint.
2. Make your change and click **Update**.

**To delete an entry:**

- Click **Delete** under Action for the endpoint.

### Choosing Allowable SAML Bindings (SAML 2.0)

At this step for SAML 2.0 connections, you select the binding(s) that your SP partner will use to send SAML authentication requests or SLO messages.

This configuration applies to SP-initiated SSO and to either SLO profile.

### Setting an Artifact Lifetime (SAML)

When you send an artifact to your SP’s Assertion Consumer Service or SLO service (for SAML 2.0), an element in the message indicates how long it should be considered valid.

You can change the default value per your requirements, if needed. Also consider synchronizing clocks between your server and your partner's SAML gateway server. If clocks are not synchronized, you might need to set the artifact lifetime to a higher value.

**To reach this screen for editing:**

1. Click a connection name on the **Main Menu**.
   - Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the SP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **Protocol Settings** on the Summary screen.
5. Click **Configure Protocol Settings**.
6. Click **Allowable SAML Bindings** on the Summary screen.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click Artifact Lifetime on the Summary screen.

This step appears only if you have selected the artifact binding for either an SSO or SLO Service (under SAML 2.0) at the SP site.

### Specifying Artifact Resolver Locations (SAML 2.0)

This endpoint or group of endpoints is where your server will send back-channel requests to resolve artifacts received from your partner. The locations are also known collectively under SAML specifications as the Artifact Resolution Service.

To reach this screen for editing:

1. Click a connection name on the Main Menu.
2. Click Manage All SP, if needed, to see a full list of connections.
3. Click Browser SSO under the SP Connection tab.
4. Click Configure Browser SSO.
5. Click Protocol Settings on the Summary screen.
6. Click Artifact Resolver Locations on the Summary screen.

If this step does not appear, you do not have Artifact selected under Allowable SAML Bindings.

To configure the Artifact Resolver Location(s):

1. Enter a URL on the Artifact Resolver Locations screen and click Add.

The URL must be fully qualified (defining protocol, host, and port) unless you have entered a base URL (see “General Information” on page 255).

Repeat this step if your SP supports multiple services. The SAML 2.0 specifications permit multiple artifact resolution services through the use of Index numbers, which PingFederate automatically supplies when you add a service. Alternatively, if needed per partner specifications, you may assign these index numbers manually.

**Note:** When specifying multiple artifact resolution endpoints, each endpoint must share the same transport protocol. That is, if one endpoint uses HTTP, then all must use HTTP. Similarly, if one endpoint uses HTTPS, then all must use HTTPS.
2. Click **Next**.

**Defining Signature Policy**

The Signature Policy screen provides options controlling how digital signatures are used for SSO Internet messaging. The choices made on this screen depend on your partner agreement (see “Digital Signing Policy Coordination” on page 28).

Digital signing is required for SAML Response messages sent from your site via POST (or Redirect for SAML 2.0). Optionally, SSO authentication requests from the SP (SP-initiated SSO) may also be signed to enforce security. (This option appears only for SAML 2.0 connections and only if you have enabled SP-initiated SSO using the POST or redirect bindings.)

The assertions inside SAML Responses may be also be signed. When you make this choice, only the assertion portion of the Response is signed, not the complete Response. (This is the only option that appears for SAML 1.x connections.)

- Make your selection(s) and click **Next**, or just click **Next** if no additional security is required.

**Configuring XML Encryption Policy (SAML 2.0)**

For SAML 2.0 configurations, in addition to using signed assertions to ensure authenticity, you and your partner may also agree to encrypt all or part of an assertion to improve privacy. This feature is commonly used if the assertion might pass through an intermediary (such as a user's browser) and HTTPS is not used.

If the name identifier (or `SAML_SUBJECT`) of an assertion is encrypted, you and your partner may also want to encrypt the identifier in subsequent single-logout messages (if you are using an SLO profile).

Note that “The entire assertion” selection on the Encryption Policy screen includes the `SAML_SUBJECT` and all attributes.
To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Browser SSO under the SP Connection tab.
3. Click Configure Browser SSO.
4. Click Protocol Settings under the Browser SSO tab.
5. Click Configure Protocol Settings.

To define XML encryption:
1. Choose whether you want to encrypt the entire assertion or one or more attributes.
2. If you are encrypting the name-identifier attribute, use the checkboxes near the bottom of the screen to indicate whether you will also encrypt this attribute in outbound SLO messages and/or allow its encryption for inbound messages.
3. Click Next or Done.

To disable previously configured XML encryption selections:
1. Select None and then Done.
2. Click Save on the Protocol Settings screen.

Editing and Saving Protocol Settings
On the Summary screen you can review or edit your Protocol Settings.

Important: When you finish editing existing settings, be sure to click Done on the Summary screen and then Save on the Protocol Settings screen. For a new connection, click Done and then click Next on the Protocol Settings screen. Save the entire connection on the Activation & Summary screen (see "Editing and Activating a Connection" on page 328).

To reconfigure saved settings:
1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.
   If you need to make dependent or other changes, do so and continue by clicking Done until you reach the Protocol Settings screen.
3. Click Save on the Protocol Settings screen.
Editing and Saving Browser SSO Settings

On the Summary screen for Browser SSO, you can review or edit your SSO configuration.

**Important:** When you finish editing existing settings, be sure to click Done on the Summary screen and then Save on the Browser SSO screen. For a new connection, click Done and then click Next on the Browser SSO screen. Save the entire connection on the Activation & Summary screen (see “Editing and Activating a Connection” on page 328).

**To reconfigure saved settings:**

1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.
   
   If you need to make dependent or other changes, do so and continue by clicking Done until you reach the Browser SSO screen.
3. Click Save on the Browser SSO screen.

Configuring the Attribute Query Profile

At the Attribute Query step you configure your connection to respond to requests for user attributes from your partner SP, if you have chosen this option (see “Choosing Connection Options” on page 254). Attribute queries are not dependent on single sign-on but may be used independently or in conjunction with Browser SSO or provisioning to provide flexibility in how a user authenticates with SP applications (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

To continue, click **Configure Attribute Query Profile**.

Defining Retrievable Attributes

On this screen you specify the user attributes you and your partner have agreed to allow in an attribute query transaction. Note that the SP may not necessarily request all of these attributes in each attribute-query request. Instead, the list simply limits the request to a subset of these attributes.
To add an attribute:

- Enter the attribute name in the text box and click **Add**.

To edit an attribute name:

1. Click **Edit** and make your change.
2. Click **Update**.

To delete an attribute:

- Click **Delete**.

### Configuring Attribute Lookup

Attribute sources are specific data store or directory locations containing information that may be returned to the SP in response to an attribute request.

This portion of the attribute query configuration allows you to configure one or more data stores to look up attributes and to set up search parameters.

To configure an attribute source:

- Click **Add Attribute Source** and complete the setup steps (see "Choosing a User-Data Store" on page 301 next).

To modify an attribute source configuration:

1. Click the attribute source Description link.
2. Click **Save** on the screen you change.

**Note:** Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click **Save** or **Done** when either of those options appears.
To reach this screen for editing:
1. Click the connection name on the Main Menu.
2. Click Manage All SP, if needed, to see a full list of connections.
3. Click Attribute Query under the SP Connection tab.
4. Click Configure Attribute Query Profile.
5. Click Attribute Source & User Lookup on the Summary screen.

Choosing a User-Data Store

Because no user authentication is performed in response to an attribute-query request, you cannot use attributes drawn from the user's session (see “IdP Adapter Mapping” on page 265). Therefore, you must identify the data stores that contain the attributes on your system.

<table>
<thead>
<tr>
<th>Data Store</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Source Id</td>
<td></td>
</tr>
<tr>
<td>Attribute Source Description</td>
<td></td>
</tr>
<tr>
<td>Active Data Store: - SELECT -</td>
<td></td>
</tr>
<tr>
<td>Data Store Type</td>
<td>None</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage ALL SP, if needed, to see a full list of connections.
2. Click Attribute Query under the SP Connection tab.
3. Click Configure Attribute Query Profile.
4. Click Attribute Sources & User Lookup on the Summary screen.
5. Click an attribute source Description link.

To define an attribute source:
1. Use Attribute Source Id to uniquely identify the data source for the mapping.
2. Use Attribute Source Description to specify an attribute source name that distinguishes this user lookup for the selected data store.

**Note:** PingFederate appends this description to the data store type in the Source list on the Attribute Mapping Fulfillment screen.

3. Choose an Active Data Store and click Next.
   A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click Manage Data Stores to add it (see “Managing Data Stores” on page 116).
Configuring Data Store Lookup

The process of configuring PingFederate to look up attributes in a data store for attribute-query responses is similar to that used for SSO Attribute Sources and User Lookup. For detailed information, see the step-by-step procedures in the sections indicated below.

If you use a JDBC data store, see:
- “Selecting a JDBC Database Table and Columns” on page 273
- “Configuring a Database Filter (WHERE Clause)” on page 275

If you use a LDAP data store, see:
- “Configuring an LDAP Directory Search” on page 277
- “Configuring an LDAP Filter” on page 280

If you use a Custom data store, see:
- “Configuring Custom Source Filters” on page 282
- “Selecting Custom Source Fields” on page 282

Note that the screen text may differ slightly. In addition, note that the variable ${SAML_SUBJECT} is available on the Database or LDAP Filter screens to retrieve the subject identifier from the Attribute Query for use in data-store query statements.

Important: When attribute queries are sent using XASP, use the variable ${SubjectDN}—rather than ${SAML_SUBJECT}—to retrieve the subject identifier. You may also use any of these DN-parsing variables: $(CN), $(OU), $(O), $(L), $(S), $(C), and $(DC).

If more than one value exists for any of the parsing variables, then they are enumerated. For example, if the Subject DN is:

cn=John Smith,ou=service,ou=employee

then you could use any of these elements in your filter qualifier:

${SubjectDN}=cn=John Smith,ou=service,ou=employee
${ou}=service
${ou1}=employee

For more information about XASP, see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started.

Attribute Mapping Fulfillment

The last step in configuring an attribute source is to map values into the assertion to be sent in response to an attribute query.

You map attributes on the Attribute Mapping Fulfillment screen.
Map each attribute into the assertion from one of these Sources:

- **Context**
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **LDAP/JDBC/Custom**
  Values are returned from your attribute source. When you make this selection, the Value list is populated by the LDAP, JDBC, or Custom attributes you identified for this Attribute Source.

  **Note:** PingFederate appends a description in parentheses for each data store lookup (see “Choosing a User-Data Store” on page 301).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  This can be text only, or you can mix text with references to any of the values from your user-data store using this syntax:

  ${ds.attr-source-id.attribute}

  where **attr-source-id** is the Attribute Source Id value (see “Choosing a User-Data Store” on page 301) and **attribute** is any of the data store attributes you have selected.

  There are a variety of reasons why you might hard code a text value. For example, if your SP's Web application provides a service based on your company's name, you might provide that attribute value as a constant.

**Defining Issuance Criteria (Optional)**

Use this screen to define criteria PingFederate can evaluate to determine whether to issue an attribute query response (see “About Token Authorization” on page 24). This extension of token authorization can be used to restrict who can access protected resources.
Chapter 6
Identity Provider SSO Configuration

To configure Issuance Criteria:
1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   - Context – Select to use values returned from the context of the transaction at runtime.
   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.
   - Mapped Attributes – Select to access retrievable attributes defined for the attribute query profile.

Note: For the HTTP Request selection, because it is retrieved as an object rather than text, you must generally use OGNL expressions for evaluation. Click Show Advanced Criteria to use expression mapping. (If that button is not displayed on the screen, expressions are not enabled—see “Enabling and Disabling Expressions” on page 573). For syntax and examples, see sections under “Constructing Expressions” on page 574.)

2. Select an attribute name.
3. Select the Condition you want to apply.

Note: Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

The Error Result field is used by the StatusMessage element in the SAML response to the SP.

**Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

If you leave this field blank, the user sees a default ACCESS_DENIED error result at runtime if the authorization fails.

6. Click **Add**.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

- Use the in-line editor box to enter the OGNL expression.
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

**Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click **Add**. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**

- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**

- Click **Delete** under Actions for the criteria and then click **Save**.
Specifying Security Policy

This screen allows you to specify the digital signing and encryption policy to which you and your partner have agreed. These selections will trigger requirements for setting up Credentials (see “Configuring Credentials” on page 306).

To configure attribute-query security policy for this partner:

► Check or clear the checkboxes and click Next or Done.

Editing and Saving Attribute Query Configurations

To reconfigure saved profiles:

1. Click the heading over the information you want to change.

2. Click Done on the screen containing your change.

   If you need to make additional changes, do so and continue by clicking Done until you reach the Attribute Query screen.

3. Click Save on the Attribute Query screen.

Configuring Credentials

The Credentials screen presents a list of possible security requirements you might need, depending on the federation protocol you are using and the choices you have made.

► To continue, click Configure Credentials.
Managing SP Connections

Configuring Back-Channel Authentication

When you configure a profile for inbound SAML messages via the artifact binding, you must specify authentication information for outbound artifact resolution requests over SOAP to your SP’s Artifact Resolution Service.

Similarly, if you configure outbound Assertion Consumer Service or SLO Service URLs to use the artifact binding, then you must configure SOAP authentication requirements for inbound messages such as artifact resolution requests. If you configure outbound SLO Service URLs to use the SOAP binding, then you must also configure authentication requirements for outbound SOAP messages.

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Credentials under the SP Connection tab.
3. Click Configure Credentials.
   If this step is not present, then it is not applicable to your configuration—you have not configured any profiles that use an artifact or SOAP binding or allowed artifact as an inbound SAML binding.

To configure back-channel authentication requirements for sending SOAP messages:
1. On the Back-Channel Authentication screen, click the Configure link to the right of the list of messages to be sent to your partner.
2. Make one or more selections on the Outbound SOAP Authentication Type screen:
   - HTTP Basic — you enter SOAP Basic credentials on a later screen.
   - SSL Client Certificate — you specify the certificate on a later screen.
     This option is available only if you specify an endpoint that uses SSL.
   - Use Digital Signatures (Browser SSO profile only) — you sign the message.
     You are asked to select a signing certificate on a later screen.

For SAML 2.0, these options may be used in any combination or independently. For SAML 1.x, you must use either Basic or SSL authentication; digital signing may be added to ensure message integrity.

By default, PingFederate validates your partner’s SSL server certificate—verifying that the certificate chain is rooted by a trusted Certificate Authority.
and that the hostname matches the certificate's Common Name. Clear the associated checkbox if you do not want this validation to occur.

3. Click **Next**.

4. If you chose **HTTP Basic** at Step 2, enter the SOAP Username and Password to use for this partner under Basic SOAP Authentication.

   You must obtain these credentials from your partner.

5. If you are using an SSL certificate, select the certificate under SSL Authentication Certificate and click **Next**.

   If you have not yet created or imported the client SSL certificate you need into PingFederate, click **Manage Certificates** (see “SSL Client Keys and Certificates” on page 207). You need to export the certificate (only) and send it to your partner.

6. On the Summary screen, click **Done**.

**To configure back-channel authentication requirements for receiving SOAP messages:**

1. On the Back-Channel Authentication screen, click the **Configure** link to the right of the list of messages to be received from your partner.

2. Select one or more options on the Inbound Authentication Type screen:
   - **HTTP Basic** — Enter the logon username and password your partner uses on the next screen.
   - **SSL Certificate** — Specify certificate verification information on a later screen.
   - **Use Digital Signatures (Browser SSO profile only)** — Incoming messages must be signed.
   - **Require SSL** — When selected, incoming HTTP transmissions must use a secure channel.

   You are asked to select a signature verification certificate on a later screen. For SAML 2.0, use these options in any combination or independently. For SAML 1.x, you must use either Basic or SSL authentication; add digital signing to ensure message integrity.

3. Click **Next**.

4. If you chose **HTTP Basic** at Step 2, enter the **Username** and **Password** under Basic Authentication (Inbound).

   **Important:** If you are configuring more than one connection that uses the artifact or HTTP profile, you must ensure that the **Username** is unique for each connection.

5. If you are using an SSL certificate, select Anchored or Unanchored under Certificate Verification Method.
   - **Anchored** — The certificate must be signed by a trusted Certificate Authority, and the CA's certificate must be imported into the PingFederate Trusted CA store (see “Trusted Certificate Authorities” on page 204).
Managing SP Connections

- Unanchored — The certificate is self-signed or you want to trust a specified certificate.

**Note:** When anchored certificates are used between partners, certificates may be changed without sending the update to your partner. If the certificate is unanchored, any changes must be promulgated.

6. Click **Next**.

7. If you chose anchored SSL certificate verification at Step 5, enter the Subject DN and click **Next**.

**Tip:** If you have not yet defined the certificate in PingFederate or you do not know the DN, return to the previous screen and select Unanchored. Then click **Next** and click **Manage Certificates** on the SSL Verification Certificate screen to import the certificate, if needed, or to view its DN.

8. If you chose unanchored SSL certificate verification at Step 5, select the certificate to use for validating the SSL connection.

If you have not yet imported the certificate into PingFederate, click **Manage Certificates**.

9. Click **Next**.

10. On the Summary screen, click **Done**.

**Configuring Digital Signature Settings**

This step defines the private key/certificate that you will use to sign assertions and SLO messages for this SP.

**Note:** Digital signing is required for SSO assertions and SLO messages sent via POST or redirect bindings. Signing is not always required for profiles using the artifact or SOAP bindings.

The step applies to both IdP- and SP-initiated SSO and to either SLO profile (see “Choosing Profiles (SAML 2.0)” on page 259) whenever outbound POST or redirect bindings are used. The step also is required for WS-Trust STS and for SSO if you chose to sign the SAML assertion, SAML response, or artifact resolution messages (see “Configuring Back-Channel Authentication” on page 307).
Chapter 6
Identity Provider SSO Configuration

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Credentials under the SP Connection tab.
3. Click Configure Credentials.
4. Click Digital Signature Settings on the Summary screen.

To specify a certificate:
1. Select the certificate from the drop-down list.
   If you have not yet created or imported your certificate into PingFederate, click Manage Certificates (see “Digital Signing and Decryption Keys and Certificates” on page 209).
2. (Optional) If you have agreed to send your public key with the SAML message, select the checkbox to include the certificate. To include the raw key in the signature as well, select the “Include the raw key …” checkbox.

   Note: For WS-Trust STS connections, by default the <KeyInfo> element in the SAML token includes a reference to the certificate rather than the full certificate unless this box is checked.

3. (Optional) Select the Signing Algorithm from the drop-down list.
   The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

Configuring Signature Verification Settings

Under SAML 2.0 specifications, when your site receives any SAML 2.0 messages via the POST or Redirect bindings, the messages must be digitally signed. Signing is also always required for the SAML 1.x POST binding and for WS-Federation assertions, as well as incoming SAML 1.1 or 2.0 tokens for WS-Trust STS processing.

Depending on your agreement with this SP, SSO assertions, SAML 2.0 artifacts, or SOAP messages might also require signatures.

Whenever signatures are required, PingFederate provides a choice of trust models, including an option to use anchored signature-verification certificates embedded in incoming messages (see “Trust Models” on page 28). When this option is chosen in Signature Verification Settings, you must provide the Subject DN for embedded certificates coming from this partner, and the Issuer CA certificate must be part of the PingFederate trusted store (see “Trusted Certificate Authorities” on page 204).

Alternatively, you may choose to use unanchored certificates, in which case you must import your partner's public-key certificate during this configuration (or select it if it is already imported). To prevent any interruption of service due to an expired certificate, you can ask your partner for a new certificate in advance and import it as backup.
To continue, click **Manage Signature Verification Settings**.

**To reach this screen for editing:**

1. Click a connection name on the **Main Menu**.
   
   *Click **Manage All SP**, if needed, to see a full list of connections.*

2. Click **Credentials** under the **SP Connection** tab.

3. Click **Configure Credentials**.

4. Click **Signature Verification Settings** on the **Summary** screen.

   *If this step does not appear, then your configuration does not require verification settings.***

**Choosing a Trust Model**

This screen allows you to choose the **Trust Model** you want to use for signature verification (see “Trust Models” on page 28).

**Important:** If you are using the Redirect binding for SLO, you cannot use anchored certificates because SAML 2.0 does not permit certificates to be included using this transport method.

**To reach this screen for editing:**

1. Click a connection name on the **Main Menu**.

   *Click **Manage All SP**, if needed, to see a full list of connections.*

2. Click **Credentials** under the **SP Connection** tab.

3. Click **Configure Credentials**.

   *For **Anchored**, the next step is to enter the Subject DN for your partner's certificate (see the next section, “Anchored Certificates--Specifying a Subject DN”).*

   *For **Unanchored**, the next step is to import your partner's certificate (see “Selecting Unanchored Certificates” on page 312).***

   **Important:** If you are using the Redirect binding for SLO, you cannot use anchored certificates because SAML 2.0 does not permit certificates to be included using this transport method.

   *For **Unanchored**, the next step is to import your partner's certificate (see “Selecting Unanchored Certificates” on page 312).***
4. Click **Signature Verification Settings** on the Summary screen. If this step does not appear, then your configuration does not require verification settings.

5. Click **Manage Signature Verification Settings**.

6. Click **Trust Model** on the Summary screen.

**Anchored Certificates--Specifying a Subject DN**

When you choose to use an anchored certificate for signature verification, incoming SAML messages must contain the partner's verification certificate (see “Trust Models” on page 28). PingFederate verifies that the Issuer CA is trusted and checks to see that the embedded certificate’s Subject DN matches the one specified on this screen. If so, PingFederate uses that certificate to verify the message signature.

You can either enter the Subject DN or extract it from your partner’s certificate if the certificate is stored on an accessible file system.

**Important:** Ensure that you enter the Subject DN correctly or extract the DN from the correct partner certificate.

To extract the Subject DN from a certificate:

1. Click browse to select your SP partner’s public certificate.
2. Click **Extract**.

To reach this screen for editing:

1. Click a connection name on the Main Menu.
2. Click **Manage All SP**, if needed, to see a full list of connections.
3. Click **Credentials** under the SP Connection tab.
4. Click **Configure Credentials**.
5. Click **Signature Verification Settings** on the Summary screen.
6. Click **Manage Signature Verification Settings**.

**Selecting Unanchored Certificates**

On the Signature Verification Certificate screen, you identify your partner’s imported public certificate and, optionally, a secondary certificate to use when the first expires (see “Trust Models” on page 28).
To reach this screen for editing:

1. Click a connection name on the Main Menu.
   Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Credentials** under the **SP Connection** tab.
3. Click **Configure Credentials**.
4. Click **Signature Verification Settings** on the Summary screen.
   If this step does not appear, then your configuration does not require verification settings.
5. Click **Manage Signature Verification Settings**.
6. Click **Signature Verification Certificate** on the Summary screen.

To specify a verification certificate:

1. Select the certificate from the drop-down list.
   If you have not yet imported the certificate into PingFederate, click **Manage Certificates**.
2. (Optional) Select a Secondary certificate for backup.
   Use this field if your partner has sent you a new certificate to replace one that is ready to expire. The server will automatically verify against the secondary certificate when the primary one expires.

**Selecting an Encryption Certificate (SAML)**

To enable XML encryption of all or part of an SSO assertion, you must identify the encryption certificate you will use (see “Configuring XML Encryption Policy (SAML 2.0)” on page 297).

You must also select a certificate if your requirements include encrypting an assertion in response to an attribute query (see “Specifying Security Policy” on page 306).
Chapter 6
Identity Provider SSO Configuration

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Credentials under the SP Connection tab.
3. Click Configure Credentials.
4. Click Select XML Encryption Certificate.

To identify the encryption certificate:
1. (Optional) Change the default settings under Block Encryption Algorithm.
   Due to import control restrictions, the standard JRE distribution supports strong but not unlimited encryption. To use the strongest AES encryption, when permissible, download and install the appropriate version of “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” from the Oracle download Web site (www.oracle.com/technetwork/java/javase/downloads/index.html).
   For more information about XML block encryption and key transport algorithms, see the “XML Encryption Syntax and ProcessingW3C Recommendation” (http://www.w3.org/TR/xmlenc-core/).
   Note: As a Key Transport Algorithm, RSA-v1.5 is disabled for new connections for security reasons. If you are updating an existing connection that uses RSA-v1.5, we recommend changing the selection for increased security.
2. From the drop-down list, select the applicable certificate and click Next.
   If the certificate is not in the list, click Manage Certificates to import it.

Selecting a Decryption Key (SAML)
If SAML_SUBJECT is encrypted, either by itself or as part of a whole assertion, then all references to this name identifier in SLO requests from your partner may also be encrypted (if the connection uses SP-initiated SLO under SAML 2.0).
To enable XML encryption, you must identify a certificate for PingFederate to use to decrypt incoming SLO messages.
To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click Credentials under the SP Connection tab.
3. Click Configure Credentials.
4. Click Select XML Decryption Key.
   If this step is not present, you have chosen not to encrypt the assertion or the SAML_SUBJECT attribute (see “Configuring XML Encryption Policy (SAML 2.0)” on page 297).

To identify the decryption key:

- From the drop-down list, select the applicable certificate and click Next.
  If the certificate is not in the list, click Manage Certificates to import it (see “Digital Signing and Decryption Keys and Certificates” on page 209).

Editing and Saving Credential Configurations

From the Summary screen you can review or edit your credentials configuration.

Important: When you finish editing existing settings, you must click Done on the Summary screen and then Save on the Credentials screen. For a new connection, click Done and then click Next on the Credentials screen. Save the entire connection on the Activation screen (see “Editing and Activating a Connection” next).

Configuring Outbound Provisioning

PingFederate’s Outbound Provisioning (formerly SaaS Provisioning) allows an IdP to create and maintain user accounts at standards-based partner sites and select-proprietary provisioning partner sites that are protocol-enabled (see “Outbound Provisioning for IdPs” on page 34).

Note: This configuration task is presented in the administrative console only when you enable Outbound Provisioning (see “Choosing a Connection Type” on page 252).
To continue, click **Configure Provisioning**.

**Note:** Screen illustrations in this section are presented in most cases for service providers using **SCIM**, the provisioning plug-in bundled with PingFederate. However, the screens are comparable for SaaS vendors for whom provisioning is supported, and the information and instructions provided are the same unless otherwise noted.

### Defining a Provisioning Target

Information on the Target screen indicates the provider's Web-service endpoint for provisioning users and, if required, credentials that PingFederate uses for authentication to the provisioning API for the service provider.
To configure the Target:

1. Enter the endpoint for managing users.
2. (Optional) Enter the endpoint for managing groups.
3. Select the authentication method the endpoint accepts.
   Select None if the endpoint does not require authentication.
4. If you select either Basic Authentication or OAuth 2.0 Bearer Token as the Authentication Method, enter valid credentials (User/Password) for your provider.
   Additionally, if you choose OAuth 2.0 Bearer Token as the Authentication Method, in order to support the password grant type, enter valid credentials for Client ID, Client Secret, and Token Endpoint URL.
5. If the SCIM provider does not support PATCH updates, clear the associated checkbox.
   For information about PATCH, see the SCIM specification (www.simplecloud.info/specs/draft-scim-api-01.html#edit-resource-with-patch).
6. (Optional) Depending on the target-provider needs, select the option to provision groups by supplying complete LDAP DNs, rather than only Common Names (CNs), to identify groups.
   Some SCIM providers, including PingOne (Ping Identity’s cloud identity management service), allow administrators to parse full DNs when necessary—for example, in the case of duplicate CNs—to determine group access mapping to specific applications based on other DN elements.
7. Select a Deprovision Method.
   The Deprovision Method options control the mapping to SCIM protocol messages sent to the partner. The deprovisioning is triggered by the user being removed from the monitored local identity store.
   • Delete User removes the user account.
   • Disable User deactivates the user account.
8. Click Next.
Managing Channels

A provisioning channel is a mapping configuration between user attributes contained in a source user store and attributes supported or required by the targeted software-service application. You can have multiple channels to the same target as needed—for example, if your organization has separate LDAP stores (or different nodes in the same store) for various user groups needing SSO access and provisioning to the same domain.

Tip: There can be only one target domain for provisioning per connection (“Defining a Provisioning Target” on page 316). If your organization subscribes to multiple domains for which you need provisioning support, you need a separate SP connection for each domain.

Channels are created and managed from the Manage Channels screen.

To access the Manage Channels screen:
1. In the task headings for a connection, click Outbound Provisioning.
   If this task is not present, provisioning is not enabled (see “Choosing Roles and Protocols” on page 106).
2. On the Outbound Provisioning screen, click Configure Provisioning.

To configure a new channel:
1. Click Create and follow the configuration steps.

Tip: If you are creating a second channel to a vendor, you can copy an existing channel and make the necessary changes.

To copy a channel:
1. Click Copy under Action for the channel you want to copy.
2. Enter new General Info for the channel (see “Specifying Channel Information” on page 319).
3. Make any further changes needed for the new channel.

To edit a channel:
1. Click the Channel Name link.
Managing SP Connections

To delete a channel:

1. Under Action, click **Delete** for the channel.
   (To undo the deletion, click **Undelete**.)

   **Note:** The **Delete** function is not available if the channel is active
   (see “Channel Activation and Summary” on page 327).

2. To confirm the deletion, click **Done** and then **Save** on the Outbound
   Provisioning screen.

Specifying Channel Information

On the Channel Info screen, specify a unique identifier for the channel.

| Channel Name | + |
| Max Threads  | 1 |

**Tip:** Adjust the Max Threads setting as needed to optimize data-
transfer performance, particularly if large numbers of records need to
be provisioned at the target site.

Identifying the Source Data Store

PingFederate fully supports Active Directory and the Oracle Directory Server as
source user repositories for Outbound Provisioning. However, you can use other types
of LDAP servers, either identifying them as Generic or registering them with
PingFederate (see “Configuring an LDAP Connection” on page 121).

Information from your user-data store is used to supply mapped values for each user
attribute required by the service provider (see “Mapping Attributes” on page 324).

On the Source screen, choose the LDAP store to use for this channel.

**Note:** If the correct data store is not listed, then it has not yet been
identified to PingFederate; click **Manage Data Stores** to set a
connection to a user store (see “Configuring an LDAP Connection”
on page 121).
Modifying Source Settings

The Source Settings screen shows the default configuration of the data store selected on the previous screen, including settings used by the PingFederate provisioner to determine when user information is added, changed, or removed.

Note: In accordance with common practice, the provisioner does not delete deprovisioned users from target data stores. Rather their status is changed to indicate that the accounts are no longer active. For some providers (such as Google Apps), default views may need to be adjusted to show deactivated user records.

If you are using the Oracle Directory Server or Active Directory, in most cases no changes are needed on this screen unless your data store uses a customized schema.

If you are using a different LDAP directory, you must supply the required information on this screen unless you have defined a template for the data store (see “Defining an LDAP Type” on page 123).
## Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entry GUID Attribute</td>
<td>The name of the attribute in the data store representing the user’s Globally Unique Identifier.</td>
</tr>
<tr>
<td>GUID Type</td>
<td>Indicates whether the GUID is stored in binary or text format. Active Directory is always binary. Other LDAP stores most often use text.</td>
</tr>
<tr>
<td>Member of Group Attribute</td>
<td>A multi-value user attribute containing the DNs of the groups to which an entry belongs. This attribute does not apply to some LDAP servers, including the Oracle Directory Server. The attribute below is used instead. Active Directory uses both values to provide a two-way mapping between User and Group objects.</td>
</tr>
<tr>
<td>Group Member Attribute</td>
<td>The name of a multi-value group attribute used to track membership in the group using either DN or GUID values.</td>
</tr>
<tr>
<td>User Object Class</td>
<td>The LDAP object class to which user entries belong, used to restrict search results to user entries only.</td>
</tr>
<tr>
<td>Changed Users Algorithm</td>
<td>The method by which PingFederate determines if user records have been updated or new records added, thus requiring provisioning updates at the target site. The three choices are:</td>
</tr>
<tr>
<td></td>
<td><strong>Active Directory USN</strong> – For Active Directory only, this algorithm queries for update sequence numbers on user records that are larger than the last time records were checked.</td>
</tr>
<tr>
<td></td>
<td><strong>Timestamp</strong> – Queries for timestamps on user records that are not older than the last time records were checked. This check is more efficient from the point of view of the PingFederate provisioner but can be more time consuming on the LDAP side, particularly with the Oracle Directory Server.</td>
</tr>
<tr>
<td></td>
<td><strong>Timestamp No Negation</strong> – Queries for timestamps on user records that are newer than the last time records were checked. This algorithm is recommended for the Oracle Directory Server.</td>
</tr>
<tr>
<td>USN Attribute</td>
<td>The name of the attribute used to store the update sequence number—applicable when the Active Directory algorithm is chosen above.</td>
</tr>
<tr>
<td>Timestamp Attribute</td>
<td>The name of the attribute used to store the timestamp on user records.</td>
</tr>
</tbody>
</table>
# Chapter 6

## Identity Provider SSO Configuration

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Status Attribute</td>
<td>The name of the attribute in which the user’s account status (active or inactive) is stored, for example, Active Directory = <code>userAccountControl</code> and Oracle Directory Server = <code>nsaccountlock</code>.</td>
</tr>
</tbody>
</table>
| Account Status Algorithm   | The method by which PingFederate determines a user’s account status. The values are:  
  - **Active Directory Bitmap** – For Active Directory, which uses a bitmap for each user entry. For more information about `userAccountControl` flags, see Microsoft’s knowledge base [here](http://support.microsoft.com/kb/305144).  
  - **Flag** – For Oracle Directory Server and other LDAP directories that use a separate attribute to store the user’s status. When this option is selected, the Flag Comparison Value and Flag Comparison Status fields below are also used. |
| Default Status             | Indicates the user’s status if the attribute is missing.                                                                                                                                                   |
| Flag Comparison Value      | Indicates the value for the attribute (for example, `nsaccountlock`) that PingFederate expects to be returned. Used when the Account Status Algorithm is set to **Flag**.                                               |
| Flag Comparison Status     | Indicates whether the user is enabled or disabled when the flag has the value specified in the Flag Comparison Value field. Setting the value to `true` equals enabled while setting the value to `false` equals disabled.  
  For example, if the Account Status Attribute is set to `nsaccountlock`, and the Flag Comparison Value is set to `true`, and the Flag Comparison Status is set to `false`, then any users with `nsaccountlock=true` are disabled.  
  Used when the Account Status Algorithm is set to **Flag**. |
Specifying a Source Location

Indicate on the Source Location screen where PingFederate should look for user records in the data store. The same location may be used to retrieve user-group DN for maintaining corresponding groups at the service provider.

**Note:** Group provisioning is supported for SCIM and the Google Apps Connector (version 2.0 and higher) but may not be supported for other SaaS Connectors. If not, the associated fields on the Source Location screen are grayed out. Support for the feature may become available in future Connector releases; contact Ping Identity Support (pingidentity.com/support-and-downloads/support-request-form.cfm) for more information.

After specifying the required Base DN, you have two options for indicating which users and user groups (optionally) are provisioned. In each case, the search can be based either on group membership (the containing Group DN in the case of user groups) or on a specified LDAP filter.

<table>
<thead>
<tr>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base DN</strong></td>
</tr>
<tr>
<td>The base Distinguished Name of the tree structure where user records and user groups are stored. PingFederate looks only at this node level or below it for user accounts and groups that need to be provisioned.</td>
</tr>
</tbody>
</table>
Chapter 6
Identity Provider SSO Configuration

Mapping Attributes

The Attribute Mapping screen provides a means of managing how attributes from your user store are mapped to SCIM attributes or to the provisioning fields supported for your organization’s SaaS-customer account.

Important: If you are provisioning for SCIM, your SP may make one or more optional core attributes mandatory. Refer to your SP’s SCIM documentation or SCIM Resource Schema representation. For SaaS Connectors, refer to connection-template setup steps in your SaaS Connector Quick Connection Guide for information about any special fields and how to map them.

Tip: For non-SCIM SaaS connectors, PingFederate automatically retrieves from the vendor the Field Names shown on this screen, but only on the first pass through the screen flow. If you are using this screen to modify an existing mapping configuration, click Refresh Fields near the bottom of the screen to synchronize the list with the target data store if needed.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| Group DN   | For Users, the group Distinguished Name associated with the user store, if applicable—required if a Filter is not used (see below). (Optional) For Groups, the DN of the higher-level group that contains the user groups, if applicable. Important: If a Group DN is used for Active Directory Groups, the domain functional level must be set as follows:  
  • For Windows Server 2012 – Any of the above or Windows Server 2012. (Refer to Windows Server documentation or support for more information.) |
| Filter     | An LDAP search filter—for Users, required if a Group DN is not used; optional for Groups. For information about LDAP filters, refer to your LDAP documentation. Note that you may need to escape any special characters. |
To configure attribute mapping:

1. Click Edit under Action for a field.
2. On the Specify Attribute Mapping screen, provide mapping details.
   (For more information, see the next section, “Specifying Mapping Details”.)
3. Repeat for each attribute shown in the Field Name column as needed.

For each field, the screen provides a means of adding or modifying the mapping details (see the next section, “Specifying Mapping Details”).

Note: All required attributes listed in the Field Name column, indicated with asterisks, must be mapped. Click View Partner Field Specifications near the bottom-left of the screen for a summary of requirements for all fields specified for the target partner.

For some fields,PingFederate preselects LDAP attributes commonly used to store the required values.

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>ATTRIBUTE(S)</th>
<th>DEFAULT</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username*</td>
<td>- sAMAccountName</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Formatted Name</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Family Name</td>
<td>- sn</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Given Name</td>
<td>- givenName</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Middle Name</td>
<td>- middleName</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Honoric Prefix</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Honoric Suffix</td>
<td>- generationQualifier</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Display Name</td>
<td>- displayName</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Nickname</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Profile URL</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Title</td>
<td>- title</td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>User Type</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Preferred Language</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Locale</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Timezone</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
<tr>
<td>Password</td>
<td></td>
<td></td>
<td>Edit</td>
</tr>
</tbody>
</table>
Chapter 6
Identity Provider SSO Configuration

Specifying Mapping Details

On this screen, you define specific mapping information for each field required for provisioning (and for any optional fields, as needed).

Tip: For most fields, if you need to map more than one attribute from your data store into a single field at the target location, then you must use an OGNL expression to indicate how the attribute values are to be combined. The use of OGNL expressions may not be enabled for your PingFederate installation (see “Using Attribute Mapping Expressions” on page 573).

The only exception is a field called LDAP Attributes Map, provided primarily to support PingOne-specific SCIM attributes. This field may contain multiple attributes without using OGNL.

Specifying Mapping Details

On this screen, you define specific mapping information for each field required for provisioning (and for any optional fields, as needed).

To define mapping information for an attribute:

1. (Optional) Select the Root Object Class containing a user-store attribute that you want to map to the provisioning attribute shown under Field Name.

   Note: For some fields, you may not need to map specific user attributes. If so, supply a Default Value instead—skip this step and go to Step 5. You can also do both for certain attributes, as needed: that is, specify LDAP attributes as well as a Default Value.

2. Under Attribute, select an attribute from the class.

Caution: If end-users at your site are permitted to edit some of their own attributes directly in the LDAP store, ensure that the attributes are restricted and do not include any needed by the service provider to grant permissions.
3. Click **Add Attribute**.

4. Repeat the steps above to add additional applicable attributes, as needed, to use in a mapping expression.

**Important:** You must add an attribute for it to be used in an expression.

5. Under Value Definition, enter or select a Default Value (optional, if one or more attributes is specified above).

A drop-down list appears for this field if the vendor requires a choice among specified values. When an expression is also supplied, the default value is sent during provisioning if an error occurs evaluating the expression.

6. If more than one attribute is used for mapping fields other than LDAP Attributes Map, enter an Expression.

**Tip:** Click **Edit** to create and validate the expression.

For information about the expression language supported by PingFederate, OGNL, see “Using Attribute Mapping Expressions” on page 573.

**Important:** The use of OGNL expressions may not be enabled for your PingFederate installation (see “Enabling and Disabling Expressions” on page 573).

7. (Optional) Select one or more processing Options, as defined below:

- **Create Only** – The field is provisioned only once and not subsequently updated.

  **Note:** For SCIM, the Password attribute should be passed only when creating a user or updating the password. Select **Create Only** to limit when the Password attribute is passed.

- **Trim** – Removes any white space from the attribute value(s).

- **Upper Case/Lower Case/None** – Transforms the attribute value(s) to the case indicated, unless **None** is selected (the default).

- **Parsing-->Extract CN from DN** – For attributes in the form of a Distinguished Name (for example, Group DNs in Active Directory), maps only the Common Name portion of the DN.

- **Parsing-->Extract Username from Email** – For attributes containing an email address, maps only the username.

**Tip:** For SaaS Connectors not bundled with PingFederate, refer to your SaaS Connector **Quick Connection Guide** for instructions on mapping options or requirements for particular provisioning fields.

### Channel Activation and Summary

When you finish setting up a channel, you may choose to activate it immediately; or you can return to the Activation & Summary screen and activate the channel when needed. Note that the overall SP connection, when applicable, also must be active.
for any provisioning channels to be enabled (see “Editing and Activating a Connection” on page 328).

**Caution:** When a channel is activated, initial provisioning occurs as soon as the synchronization-frequency time period expires (see “Configuring Outbound Provisioning Settings” on page 113). The default is 60 seconds. Initial provisioning can consume considerable processing time, depending on the amount of data that needs to be transmitted; administrators may wish to plan accordingly.

**Important:** Regardless of whether you choose to activate a new channel immediately or later, if you want to save the channel configuration, click **Done** on the Summary screen and then **Save** on the connection Activation and & Summary screen. (For a new channel in an existing connection, click **Save** on the Outbound Provisioning screen.)

You can deactivate a channel at any time (for maintenance, for example). When a channel is inactive, SSO/SLO transactions may still occur (if an associated connection is active), but provisioning is suspended.

**To change a channel status:**

- Select either **Active** or **Inactive** and then click **Done**.

**To modify a channel setting:**

- If you know which step needs to be modified, click its link under the Channel tab.

  If you do not know where to change a setting, locate the currently configured setting under one of the summary headings and then click the subheading above the information.

**Editing and Activating a Connection**

When you finish setting up a connection, you may choose to activate it immediately.

**Important:** Regardless of whether you choose to activate a new connection now or later, you must click **Save** on the Summary screen for a new connection if you want to keep the configuration.

You can deactivate a connection at any time (for maintenance, for example). When a connection is inactive, all SSO or SLO transactions to or from this partner are disabled, as well as access to the WS-Trust STS for Web Service Clients associated with this connection.

**Tip:** The SSO Application Endpoint near the top of the Summary screen is an example URL that webmasters or Web application developers at your site might use to invoke SSO for the connection. For details about SSO and other server endpoints, including optional query parameters, see “Application Endpoints” on page 529.
Defining SP Affiliations

To change a Connection Status:
▶ Select either Active or Inactive and then click Save.

To modify a connection setting:
1. If you know which step needs to be modified, click its link under the SP Connection tab.
   If you do not know where to change the setting, locate the currently configured data under one of the summary headings and then click the subheading above the data.
2. Change the information on the step screen and click Save, if available.
   If Save is not available, you are in the middle of a task (see “About Tasks and Steps” in the “Console Navigation” chapter of Getting Started); click Next or Done until you reach a screen containing a Save button. Then click Save and continue as needed until you return to the Main Menu.
   If your modification requires related configuration changes, PingFederate provides error messages indicating the necessary steps and then guides you to the related screens (unless you click Cancel).

Important: Be sure to click Save whenever that button appears, if you want to keep your changes.

Defining SP Affiliations

An SP affiliation is a SAML 2.0 specification that permits a group of service providers to make use of the same persistent name identifier for account linking (see “Account Linking” on page 17).

This may be of use when multiple service providers share a business relationship in which users need services from each affiliated provider. By agreement among the affiliation members, the same pseudonym can be used to populate the SAML_SUBJECT of assertions sent to all of the SP partners contained in this affiliation.

Important: Each connection in the affiliation must be configured to use the same IdP adapter instance for generating account links (see “IdP Adapter Mapping” on page 265).

You can create or modify an SP affiliation from the Main Menu or from a list of affiliations (click Manage All Affiliations).

To create an SP affiliation:
▶ Click Create New under SP Affiliations on the Main Menu.
   Or:
▶ Click Manage All Affiliations and then click Create Affiliation on the Select an Affiliation screen.

To delete an affiliation:
1. Click Manage All Affiliations under SP Affiliations on the Main Menu.
2. Click Delete under Action for the affiliation you want to delete.
3. Click Save to confirm the deletion (or click undelete).
To view or modify an affiliation:

- Click the affiliation name, or click Manage All Affiliations if the ID does not appear.

Using the Manage Affiliations Screen

You can manage SP affiliations on this screen.

To reach this screen for editing:

- Click Manage All Affiliations under SP Affiliations on the Main Menu.

To begin creating a new affiliation:

- Click Create Affiliation (see the next sections for more information).

To delete an affiliation:

1. Click Delete under Action for the affiliation you want to delete.
2. Click Save to confirm the deletion (or click undelete).

To view or modify an affiliation:

- Click the affiliation ID.

Importing Affiliation Metadata

An IdP may send a metadata file containing information that automatically specifies members of an SP affiliation for use in PingFederate.

- If you do not have a metadata file, click Next.

To import metadata:

1. Click Browse to locate and import the file and then click Next.
2. Review the information on the Create Affiliation page (see the next section).
3. Click Save on the Summary screen.

Entering Affiliation Information

Enter or modify basic information about an affiliation on the Affiliation General Info screen.

If you imported a metadata file, this information is already supplied. However, you may change the Affiliation ID or select a different Affiliation Owner, if required.
Defining SP Affiliations

Managing Affiliation Membership

On the Affiliation Membership screen, you create and manage a list of SP connections to be included in the affiliation.

If you imported a metadata file, this information is already supplied. However, you may add or remove connections from the affiliation.

- To add an SP partner connection to the affiliation, select the connection from the drop-down list and click Add.

> **Important:** Each connection in the affiliation must be configured to use the same IdP adapter instance for generating account links (see “IdP Adapter Mapping” on page 265).

- To remove a member of the affiliation, click Delete under Action for the connection and click Save.

> **Note:** If you delete an affiliation member supplied by an imported metadata file and then save the affiliation, that connection will not appear in the drop-down list for re-adding in the future.

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affiliation ID</td>
<td>A unique identifier for this affiliation. This value serves as the Name ID qualifier for SAML assertions sent to affiliated SP partners.</td>
</tr>
<tr>
<td>Affiliation Owner</td>
<td>Any SAML 2.0 SP connection may serve as the Owner.</td>
</tr>
</tbody>
</table>
Activating and Editing the Affiliation

From the Affiliation Management Summary screen you can activate or deactivate an SP affiliation. You also save new affiliations on this screen, or you can click heading links to go back and modify information.

To change an Affiliation Status:

- Select either Active or Inactive and then click Save.

**Important:** Be sure to click Save. Otherwise, the status will not be changed.

To edit a connection:

1. Click the heading above the information you want to modify.
2. Make your change and click Save.

Configuring SP Auto-Connect

When your SP partner is also using PingFederate 5 or higher (or is otherwise able to provide interoperable SAML 2.0 metadata via HTTP on demand), you may choose to use Auto-Connect for that partner (see “Using Auto-Connect” on page 30). This configuration can be shared among an unlimited number of SAML 2.0 partners.

**Note:** You enable the SAML 2.0 Auto-Connect profile under System Settings (see “Choosing Roles and Protocols” on page 106).

Once Auto-Connect is enabled on your PingFederate server, complete the configuration from the Main Menu under IdP Configuration. This configuration entails:

- Setting up a common connection for all Auto-Connect partners
- Establishing a list of SP partner domains authorized to use the connection

**Initial Setup**

The basic configuration for SP Auto-Connect requires only:

- Defining a period of validity for assertions (assertion lifetime)
- Choosing a signing certificate for assertions and other SAML messages
- Configuring assertion-creation information

All other partner-connection specifications are handled automatically at runtime.

**Specifying an Assertion Lifetime**

Identity-federation standards require a window of time during which an assertion is considered valid. Each assertion has a time-stamp XML element as well as elements indicating the allowable lifetime of the assertion (in minutes) before and after the assertion time stamp.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minutes Before</td>
<td>The amount of time before the assertion was issued during which it is to be considered valid.</td>
</tr>
<tr>
<td>Minutes After</td>
<td>The amount of time after the assertion was issued during which it is to be considered valid.</td>
</tr>
</tbody>
</table>

To change the default times:

- (Optional) Edit the desired setting(s) and click **Next** or **Save**.

Choosing a Signing Certificate

For Auto-Connect runtime processing, assertions and SLO messages must be signed, since they are sent over either the POST or redirect bindings (see “SAML 2.0 Profiles” in the “Supported Standards” chapter of *Getting Started*).

**Note:** The signing certificate is embedded in your server’s Auto-Connect metadata (see “Using Auto-Connect” on page 30); there is no need to exchange certificates with your partners.

You can use the same certificate used for signing metadata (see “Configuring Auto-Connect Metadata Signing” on page 114). If you use a different certificate, ensure that it meets Auto-Connect validation requirements (see “Auto-Connect Security Model” on page 33).

**To specify a certificate:**

1. Select the certificate from the drop-down list.

   If you have not yet created or imported your certificate into PingFederate, click **Manage Certificates** (see “Digital Signing and Decryption Keys and Certificates” on page 209).
2. (Optional) Select the Signing Algorithm from the drop-down list.

The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

**Configuring Assertion Creation**

Configuring assertion creation for Auto-Connect is similar to configuring the same settings for regular partner connections.

- Click **Configure Assertion Creation** to continue.

  For configuration information, refer to sections under “Assertion Creation” on page 261.

**Auto-Connect Activation and Summary**

When you finish configuring your SP Auto-Connect initial setup, you may choose to activate the common connection immediately on the Activation & Summary screen. (No runtime processing occurs until your partner’s Auto-Connect gateway is also established and a user initiates an SSO or SLO event.)

---

**Important:** Regardless of whether you choose to activate a newly configured connection now or later, you must click **Save** on the Activation & Summary screen if you want to keep the configuration.

You can deactivate the connection at any time (for maintenance, for example). While a connection is inactive, all SSO or SLO transactions to or from Auto-Connect partners are disabled.

**To change a Connection Status:**

- Select Active or Inactive and then click **Save**.

**To modify a setting:**

1. Locate the currently configured setting under one of the summary headings and then click the subheading above the data.

**Note:** Changes made to Auto-Connect settings will be out of sync, temporarily, with metadata caches that any currently active partners might be using. If your connection is in production, you might wish to lower your server’s metadata lifetime in advance of making configuration changes (see “Configuring Auto-Connect Metadata Lifetime” on page 115).
2. Change the information and click **Save**, if available.
   If **Save** is not available, additional, dependent changes are required; click **Next** or **Done** until you reach a screen containing a **Save** button. Then click **Save** and continue as needed until you return to the Main Menu.

**Specifying Allowed SP Domains**

This screen provides PingFederate with a list of trusted domain names of your Auto-Connect partners.

![Allowed SP Domains Table](image)

Normally, when PingFederate receives an authentication request from a domain in this list, the runtime engine completes the connection automatically using metadata obtained from a standard, public location—`http://saml.<domain_name>`.

(See “Using Auto-Connect” on page 30.) Alternatively, if an Auto-Connect partner elects not to use the standard location, you can supply the applicable URL.
Chapter 6
Identity Provider SSO Configuration
Service Provider SSO Configuration

In an SP role, you use the PingFederate administrative console to configure local application-integration information and to manage connections to your IdP-partner sites. You must configure Server Settings from the Main Menu to establish your site as an SP before configuring connections to IdPs (see “Choosing Roles and Protocols” on page 106).

Note that only one connection is needed per partner, even if you are integrating more than one Web application. You can configure more than one connection, however, if your partner supports multiple protocols, or supports multiple federation IDs for the same protocol (see “Federation Server Identification” on page 37).

Note: This chapter applies to configuration settings needed for browser-based SSO. While there is some cross-over information also applicable to WS-Trust STS, if you are using PingFederate exclusively as an STS, start with “WS-Trust STS Configuration” on page 437.

Under some conditions, you can enable SSO for an unlimited number of partners at once by configuring a single, common connection (see “Using Auto-Connect” on page 30).
Chapter 7
Service Provider SSO Configuration

SP Application Integration Settings

The integration of local applications with PingFederate is the essential “last-mile” configuration that allows end-users at your IdP partner’s Web site to access your protected resources. This process is facilitated through the use of application-integration kits and a robust Software Development Kit (see “SSO Integration Kits and Adapters” on page 14).

Under Application Integration Settings on the Main Menu, you configure the SP Adapters that PingFederate uses to create user sessions that allow SSO access to your protected resources. You can also set Default URLs to which users may be directed during SSO or SLO, and you can look up system endpoints that application developers at your site need to access PingFederate’s SSO/SLO services.

Note: If your PingFederate configuration enables the WS-Trust STS, the selections under Application Integration Settings also include a link for configuring plug-in Token Generators (see “Configuring Token Generators” on page 476).

Configuring SP Adapters

SP adapters are used to create a local-application session for a user in order to provide SSO access to your application(s) or other protected resources (see “SSO Integration Kits and Adapters” on page 14). You can configure multiple instances of adapters (based on one or more adapters) to accommodate the varying needs of your IdP partners.

Note: If you are configuring an OpenToken Adapter, see “Configuring the IdP OpenToken Adapter” on page 500.

If you configure more than one adapter instance, then you must map a target URL to at least one instance (see “Mapping URLs to Adapter Instances” on page 342).

SP adapter setup is available only if your server is configured as an SP (see “Choosing Roles and Protocols” on page 106).

Important: If you install a new version of an adapter JAR file after setting up connections to instances of that adapter, you might need to reconfigure those connections. To find out, click each connection that uses the adapter (see "Accessing IdP Connections" on page 354). Errors indicating reconfiguration points may be presented.
You reach this screen by clicking **Adapters** under Application Integration Settings in SP Configuration.

**To create a new adapter instance:**

- Click **Create New Instance**.

  See the next section.

**To edit an adapter instance:**

- Click the **Instance Name** link.

**To delete an adapter instance:**

1. Click **Delete** next to the **Instance Name** on the Manage SP Adapter Instances screen. (To undo the deletion, click **Undelete**.)

   **Note:** This option is inactive if the instance is referenced elsewhere, or if it is a parent to other instances. To enable deletion, click **Check Usage** to locate the configuration(s) that depend on the instance and go to each listed configuration to remove the dependencies. If the instance you want to delete is a parent, delete child instances first.

2. Click **Save** to confirm the deletion.

**Creating an Adapter Instance**

On the **Type** screen, you begin creating an instance of an adapter that PingFederate uses for creating security sessions for your applications.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Name</td>
<td>A descriptive name for the adapter instance—for example, an identity management system name.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>An internal identifier for the adapter instance. Must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Type</td>
<td>A list of deployed adapter types available for creating a adapter instance for the server. A developer typically deploys any new adapter types before an administrator sets up a connection partner.</td>
</tr>
<tr>
<td>Parent Instance (Optional)</td>
<td>A list of configured instances for this type of adapter. If applicable, select an instance that can be used as a basis for a parent/child configuration.</td>
</tr>
</tbody>
</table>

To define an adapter instance:
1. Click Adapters on the Main Menu.
2. Click Create New Instance on the Manage SP Adapter Instances screen.

To define an adapter instance:
1. Enter the Instance Name and Instance Id on the Type screen.
2. Select the Type from the drop-down menu.
   - If the adapter you need is not listed, click Visit PingIdentity.com for additional types to see if a suitable adapter is available from the PingFederate download site. You can also create your own adapter (see “SSO Integration Kits and Adapters” on page 14).
3. (Optional) Select a Parent Instance from the drop-down list.
   - If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
4. Click Next and enter information on subsequent screens for this adapter setup, as indicated in the following sections.

   Tip: The setup steps and information needed vary with the adapters deployed on your server (see “SSO Integration Kits and Adapters” on page 14). For information about configuring the adapter packaged with PingFederate, see “Configuring the SP OpenToken Adapter” on page 504.

5. Click Done on the Summary screen.
6. Click Save on the Manage SP Adapter Instances screen.

Configuring an Adapter Instance

Configuration parameters on the SP Adapter Instance screen vary according to the adapter you choose. These options are controlled by the adapter plug-in software (see “SSO Integration Kits and Adapters” on page 14).
For information about configuring the OpenToken Adapter distributed with PingFederate, see “Configuring the SP OpenToken Adapter” on page 504.

For information about configuring an adapter contained in an integration kit, locate the User Guide under Product Documentation at pingidentity.com.

Choosing Adapter Actions

Adapters can be written to perform configuration assistance or validation actions—for example, testing a connection to Active Directory. Actions may also include generation of parameters that might need to be set manually in a configuration file.

For information about actions available using the OpenToken Adapter, see “OpenToken Adapter Configuration” on page 499.

To reach this screen for editing:

1. On the Main Menu under Application Integration Settings for SP Configuration, click Adapters.
2. Click an Instance Name.
3. Click Actions (if available).

To generate a properties list:

Click Download under Action Invocation Link.

Extending Adapter Contracts

Adapters may be written with an option allowing administrators to add to the attributes required for creating usable sessions. This feature might be needed, for example, by a legacy application that requires different authentication than other applications under the same enterprise identity-management system.

To reach this screen for editing:

1. On the Main Menu under Application Integration Settings for SP Configuration, click Adapters.
2. Click an Instance Name.
3. Click Extended Contract (if available).
To add an attribute:

- **Note:** If this is a child instance, select the override checkbox to modify the configuration.

1. Enter the attribute name in the text box and click Add.
2. Click Done then click Save on the Manage SP Adapter Instances page.

**Editing and Saving SP Adapter Instances**

From the Adapter Instance Summary screen, you can reach adapter settings for editing.

**To edit the configuration:**

1. Click the heading above the information you want to change.
2. Click Save on the configuration page and on the Manage SP Adapter Instances screen.

**To save an adapter instance:**

1. Click Done on the Summary screen.
2. Click Save on the Manage SP Adapter Instances screen.

- **Note:** If this is the second adapter instance you have configured, then Save is not yet available; you must choose whether to map the new adapter instance to an application or resource URL. In this case, click Next to continue (see “Mapping URLs to Adapter Instances” next).

**Mapping URLs to Adapter Instances**

When you configure more than one SP adapter instance, you must map target URLs to at least one adapter instance. Mapping enables you to direct inbound SAML messages to the appropriate application.

For example, this mapping configuration may be necessary in an IdP-initiated SSO scenario that connects to multiple applications at your site. For transactions initiated at your site, this mapping is needed for default situations, in cases where the target and adapter instance are not specified when the SSO/SLO is started (see “SP Endpoints” on page 533). (When this information is provided with the SP request, the mapping table is ignored.)

This screen is available only if your server is configured as an SP and if you are using more than one adapter instance, or if you have previously mapped an adapter instance on this screen.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>The target URLs that align with your configured adapter instances. The URLs instruct the PingFederate SP server to route session-creation processing through an adapter instance. If the URL in the incoming request is not matched by the first entry in this table, subsequent entries are tried until a match is found. <strong>Note:</strong> If an adapter is not allowed based on restrictions imposed (see “Restricting an SP Adapter to certain virtual Server IDs” on page 373), PingFederate tries the next entry.</td>
</tr>
<tr>
<td>Adapter Instance (drop-down menu)</td>
<td>A selection of configured SP adapter instances.</td>
</tr>
</tbody>
</table>

The order of mapping is significant in that the first matching mapping, from top to bottom, determines which adapter instance receives the SAML message. For example, if two URLs are mapped in the following order:

a) http://yourapp.com/subapp/* Adapter 1

b) http://yourapp.com/* Adapter 2

The URL http://yourapp.com/subapp/start will map to Adapter 1 because it matches mapping a. If the order of the mappings were reversed, http://yourapp.com/subapp/start would map to Adapter 2 because it would find and match mapping b first.

Note that you can use only one wildcard (*) per URL.

**To reach this screen for editing:**

1. On the Main Menu under Application Integration Settings for SP Configuration, click **Adapters**.

2. Click **Map URLs to Adapter Instances**.
   - If this step does not appear, then you have created only one adapter instance (see “Configuring SP Adapters” on page 338).

**To create adapter mappings:**

1. Enter the URL and select an adapter from the drop-down menu.

2. Click **Add Mapping**.

3. Click **Save**.

**To edit adapter mappings:**

1. Click **Edit** next to the Adapter Instance. You can change the URL or select a different adapter from the drop-down menu.

2. Click **Update**.

3. Click **Save**.

**To delete adapter mappings:**

1. Click **Delete** next to the Adapter Instance.

2. Click **Save**.
   - (Click **Cancel** to abort the deletion.)
To change the order of adapter mappings:
1. Click the up or down arrows at the left to rearrange the order.
2. Click Save.

Configuring Identity Store Provisioners

PingFederate allows you to create custom Identity Store Provisioners to bridge the inbound SCIM processing of PingFederate to your own user store. For example, you might need to create a custom Identity Store Provisioner that works with an application-specific user database schema.

Using the Software Developer Kit for PingFederate, you can create and test these custom Identity Store Provisioners (see the PingFederate SDK Developer’s Guide).

Note: The Identity Store Provisioner option is active only after you enable the Inbound Provisioning protocol on the Roles & Protocol screen (see “Choosing Roles and Protocols” on page 106).

Creating an Identity Store Provisioner Instance

On the Type screen, you begin creating an instance of an identity store that PingFederate uses to bridge the inbound SCIM processing to an external user store using a custom implementation.

<table>
<thead>
<tr>
<th>Field Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Name</td>
<td>A descriptive name for the provisioner instance—for example, an identity management system name.</td>
</tr>
<tr>
<td>Instance ID</td>
<td>An internal identifier for the provisioner instance. Must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Type</td>
<td>A list of deployed provisioner types available for creating a provisioner instance for the server. A developer typically deploys any new provisioner types before an administrator sets up a connection partner. The Identity Store Provisioner Type is limited to the provisioners currently installed on your server.</td>
</tr>
</tbody>
</table>

Field Descriptions

![Currency currency] Main Manage Identity Store Provisioners Create Identity Store Provisioner Instance

![Currency currency] Type Identity Store Provisioner Extended Contract Summary

Enter an Identity Store Provisioner Instance Name and ID, select the Identity Store Provisioner Type, and a parent if applicable. The Identity Store Provisioner Type is limited to the provisioners currently installed on your server.
To define an identity store provisioner:
1. Click **Identity Store Provisioners** on the Main Menu.
2. Click **Create New Instance** on the Manage Identity Store Provisioners screen.
3. Enter the Instance Name and Instance Id on the Type screen.
4. Select the Type from the drop-down menu.
5. (Optional) Select a Parent Instance from the drop-down list. If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
6. Click **Next** and enter information on subsequent screens for this identity store setup, as indicated in the following sections.
7. Click **Done** on the Summary screen.
8. Click **Save** on the Manage Identity Store Provisioners screen.

### Defining Identity Store Provisioner Behavior

Different configuration parameters are available on the Identity Store Provisioner screen. These options are controlled by the provisioner plug-in software (see topics about identity store provisioner interfaces in the PingFederate SDK Developer’s Guide for more information).

### Extending Identity Store Provisioner Contracts

Identity Store Provisioners may be written with an option allowing administrators to add to the core attributes the plug-in instance requires. Both the core and extended contract attributes you define here must be mapped when you configure “Write Users” within an Inbound Identity Store Provisioner connection.

#### Tip:
To keep your plug-in flexible across multiple connections (assuming a one-to-one connection-to-identity store provider setup), you might want to hard code a set of “core attributes” for all connections to fulfill, and then “extend” attributes on as needed when a partner connection depends on additional attributes.
Chapter 7

Service Provider SSO Configuration

To reach this screen for editing:
1. On the Main Menu under Application Integration Settings for SP Configuration, click Identity Store Provisioners.
2. Click an Instance Name.
3. Click Extended Contract (if available).

To add an attribute:

1. Enter the attribute name in the text box and click Add.
2. Click Done then click Save on the Manage Identity Store Provisioners screen.

Editing and Saving Identity Store Provisioner Instances

From the Summary screen, you can reach identity store provisioner instance settings for editing.

To edit the configuration:
1. Click the heading above the information you want to change.
2. Make your changes.
3. Click Done on the configuration page and Save on the Manage Identity Store Provisioners screen.

To save an identity store provisioner instance:
1. Click Done on the Summary screen.
2. Click Save on the Manage Identity Store Provisioners screen.

Configuring Default URLs

As an SP, you can supply a default URL that the end-user may see when an SSO request succeeds (that is, a session is created at your site) but the target resource is not available or not specified.

Tip: You can also specify default target SSO URLs for individual IdP connections, which take precedence over this global setting (see “Configuring Default Target URLs (Optional)” on page 398).
Similarly, you can specify a default URL indicating a successful SLO to the end-user (if no other page is designated).

Your application or your partner’s application may supply these URLs at runtime (see “SP Endpoints” on page 533), but if none is provided, PingFederate will use the default values you enter on this screen unless, in the case of SSO, a default is also defined for the connection.

**Tip:** If no default targets are specified here or at the connection level (for SSO), PingFederate provides built-in landing pages for the user. These Web pages are among the templates you can modify with your own branding or other information (see “Customizing User-Facing Screens” on page 87).

### Setting Target Validation

Several SP adapters can be configured to pass security tokens or other user credentials from PingFederate to the target resource via HTTP query parameters, cookies, or POST transmittal. In all cases, these transport methods open the possibility that a third party (with specific knowledge of aspects of the IdP and/or SP network, as well as PingFederate endpoints and configuration) might be able to obtain and use valid security tokens to gain improper access to the target resource.

This potential security threat would involve using well-formed SSO links to start an SSO request for a resource at the SP site. However, the target resource designated in the link would be intended to intercept the security token by redirection to a malicious Web site.

To prevent such an attack, PingFederate provides a means of validating SSO transactions to ensure that the designated target resource exists in a domain controlled by the SP through a list of configurable URLs.

**Tip:** The per-connection default target URLs (see “Configuring Default Target URLs (Optional)” on page 398) and the SP default URL for successful SSO (see “Configuring Default URLs” on page 346) are always allowed. You do not need to enter them into the list manually.
To reach this screen:

- Click **Target Resource Validation** on the Main Menu.

**To enable target resource validation:**

1. Select the first checkbox.
2. Indicate whether to require HTTPS.

**Important:** This selection is recommended to ensure that target validation will always prevent message interception for this type of potential attack, under all conceivable permutations.
3. Enter the domain or IP address containing a target resource.
   Use the domain only, without qualifiers. For example:
   mycompany.com
   Using an initial wildcard and period for a domain name will cover multiple
   subdomains. For example:
   *.mycompany.com
   covers hr.mycompany.com or email.mycompany.com.
   (Optional) Enter the exact path of a valid target resource (case-sensitive). Starts
   with a forward slash, without any wildcard characters in the path. If left blank,
   any path (under the specified domain or IP address) is allowed. For example:
   /inbound/Consumer.jsp
   allows /inbound/Consumer.jsp but rejects /inbound/
   consumer.jsp.

   **Tip:** You can also enter multiple query parameters with or without a
   fragment.

   For example:
   /inbound/Consumer.jsp?area=West&team=IT#ref1001
   matches /inbound/Consumer.jsp?area=West&team=IT#ref1001
   but not /inbound/Consumer.jsp?area=East&team=IT#ref1001
   (Optional) Select the **Allow Any Query / Fragment** checkbox if you want to
   allow any query parameters or fragment in the target resource.

   **Note:** When selected, no query parameter and/or fragment is allowed
   in the path.

   Click **Add**.

4. Repeat the previous step to add another path for the same or another entry for a
   different domain or IP address as needed.

5. Click **Save**.

**Viewing SP Application Endpoints**

Click **Application Endpoints** on the Main Menu to see a list of endpoints and
descriptions applicable to your federation role (IdP or SP). These endpoints are built
into PingFederate and cannot be changed.

Web-application developers at your site need to know the application endpoints to
initiate transactions via PingFederate (see “SSO Integration Kits and Adapters” on
page 14).

**Note:** For specific parameters required or allowed for Application
Endpoints, see “**SP Endpoints**” on page 533.

This screen also shows a Maintenance Endpoint that you can use to verify that the
PingFederate server is running (see “**System-Services Endpoints**” on page 540).
Chapter 7
Service Provider SSO Configuration

Federation Settings

If your identity federation uses the SAML 2.0 XASP profile (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started), you may need to identify the IdP connection to which an attribute request applies. If so, click Attribute Requester Mapping under the Federation Settings section for the SP on the Main Menu.

Also under Federation Settings, you can view protocol endpoints that your federation partners need to know to access your services via PingFederate.

Attribute Requester Mapping

If you are using the XASP profile, the application(s) at your site must supply the Subject Distinguished Name (DN) to identify a user’s X.509 authentication certificate (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started). Optionally, an application may also supply an Issuer DN, which can be used to determine the correct IdP (Attribute Authority) to use for a set of users associated with an IdP.

Note: A Format query parameter must be set to a specified value for XASP (see “/sp/startAttributeQuery.ping” on page 537).

On the Attribute Requester Mapping screen, you can map X.509 identifying information to connections and specify a default connection. You reach this screen from the Main Menu under Federation Settings.

Note: The Attribute Requester Mapping link does not appear on the Main Menu unless you have enabled the SAML 2.0 protocol for the SP role (see “Choosing Roles and Protocols” on page 106). You must also select the associated XASP checkbox.

At runtime, PingFederate tries to match the certificate’s Issuer DN (if provided) against the list of Issuer DN(s), in the order shown on this screen, until a match is found. If no match is found, the server tries the Subject DN(s) in order. If no match is found, the Default connection is used.
For Issuer and Subject DNs, you can use a regular expression to match different DNs to the same connection. Only one expression may be used in any single entry. DN values must be entered in all lower-case characters.

**To map attribute requesters to connections:**
1. (Optional) Enter an Issuer DN when applicable, select a SAML 2.0 IdP Connection Name, and click **Add**.
   Repeat this step as needed for additional DNs.
2. Enter an Subject DN, select a SAML 2.0 IdP Connection Name, and click **Add**.
   Repeat this step as needed for additional DNs.
3. Select a Default IdP connection.

**To edit a mapping:**
1. Click **Edit** for the mapping in the Action column.
2. Make your changes and click **Update** in the Action column.
3. If you are editing an existing configuration, click **Save** to confirm the change.

**To reorder the mapping list:**
- Click the up or down arrow next to a DN.

**To delete a mapping:**
1. Click **Delete** for the mapping in the Action column.
2. If you are editing an existing configuration, click **Save** to confirm the deletion.

### Viewing SP Protocol Endpoints

Click **Protocol Endpoints** under Federation Settings in the SP Configuration section of the Main Menu to see a list of SAML, WS-Federation, and/or WS-Trust STS endpoints—a pop-up window displays only those endpoints related to the federation protocols enabled in Server Settings (see “Choosing Roles and Protocols” on page 106). These endpoints are built into PingFederate and cannot be changed.

Your federation partners or STS clients need to know the applicable SP Services endpoints to communicate with your PingFederate server. Configured service endpoints for SAML connections are included in metadata export files (see “Exporting Metadata” on page 57).

The table below describes each endpoint:

**Table 23: PingFederate SP Endpoints**

<table>
<thead>
<tr>
<th>Service</th>
<th>URL and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Logout Service (SAML 2.0)</td>
<td>/sp/SLO.saml2&lt;br&gt;The URL that receives and processes logout requests and responses.</td>
</tr>
<tr>
<td>Assertion Consumer Service (SAML 2.0)</td>
<td>/sp/ACS.saml2&lt;br&gt;A SAML 2.0 implementation that receives and processes assertions from an IdP. The numbers reflect the index value PingFederate uses to handle each binding.</td>
</tr>
</tbody>
</table>
Table 23: PingFederate SP Endpoints (Continued)

<table>
<thead>
<tr>
<th>Service</th>
<th>URL and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artifact Resolution Service (SAML 2.0)</td>
<td>/sp/ARS.ssaml2</td>
</tr>
<tr>
<td></td>
<td>The SOAP endpoint that processes artifacts returned from a federation partner to retrieve the referenced XML message on the back channel.</td>
</tr>
<tr>
<td></td>
<td>(See “Important” footnote in this table.)</td>
</tr>
<tr>
<td>Metadata Service</td>
<td>/</td>
</tr>
<tr>
<td></td>
<td>The default endpoint (empty path) from which partners can retrieve Auto-Connect metadata (see “Using Auto-Connect” on page 30).</td>
</tr>
<tr>
<td>Assertion Consumer Service (SAML 1.x)</td>
<td>/sp/acs.saml1</td>
</tr>
<tr>
<td></td>
<td>A SAML 1.x implementation URL that receives and processes assertions from an IdP.</td>
</tr>
<tr>
<td>Single Sign-on Service (WS-Federation)</td>
<td>/sp/prp.wsf</td>
</tr>
<tr>
<td></td>
<td>The WS-Federation implementation URL that receives and processes security tokens and SLO messages.</td>
</tr>
<tr>
<td>WS-Trust STS (two endpoints)</td>
<td>/sp/sts.wst</td>
</tr>
<tr>
<td></td>
<td>The SOAP endpoint that receives and processes SAML security-token requests from STS clients (Web Service Providers at the SP site), validating SAML tokens or validating and exchanging SAML tokens based on the configured IdP connection.</td>
</tr>
<tr>
<td></td>
<td>/pf/sts.wst</td>
</tr>
<tr>
<td></td>
<td>Initiates direct STS token-to-token exchange and token validation, from an IdP token processor to an SP token generator, when that feature is configured (see “STS Token Exchange Mapping” on page 143).</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: If multiple token-generator instances of the same type are configured for the same connection or token-to-token mapping, a query parameter must be added to either of these endpoints—see “Configuring Token Generators” on page 476.</td>
</tr>
<tr>
<td></td>
<td>(See also “Important” footnote in this table.)</td>
</tr>
</tbody>
</table>
| **Important**: If mutual SSL/TLS is used for authentication, a secondary PingFederate listening port must be configured and used by partners or STS clients for the relevant endpoints—*.ssaml* and *.wst (see “Changing Configuration Parameters” on page 74).
Managing IdP Connections

As an SP, you manage connection settings to support the exchange of federation-protocol messages (SAML, WS-Federation, or WS-Trust) with an IdP or STS client application at your site.

**Note:** If you are configuring a new connection only for WS-Trust STS, follow the sections in this part of the manual up to and including “General Connection Information” on page 360. Then turn to “WS-Trust STS Configuration” on page 437.

These settings include:

- User attributes you expect to receive in an SSO assertion (including STS SAML tokens).
- User attributes that may be requested using the Attribute Query profile (if that profile is used).
- The protocol and, for SAML, the profile you will use, including detailed security specifications (the use of digital signatures, signature verification, XML encryption, and SSL). For more information see the “Supported Standards” chapter in Getting Started.

To continue with the configuration, you and your connection partner must have decided this information in advance (see “Federation Planning Checklist” on page 36). Your federation partner must supply some connection settings and other information (see “Configuration Data Exchange” on page 39).

**Tip:** If you are configuring connections to more than one partner under SAML 2.0 specifications, or if you intend to add partners in the future, consider using Auto-Connect (see “Configuring IdP Auto-Connect” on page 434).

As an SP, you respond to user requests for SSO and SLO by creating or closing user sessions, respectively, in local applications. You integrate these applications with PingFederate by configuring them with SP adapter instances (see “Configuring SP Adapters” on page 338). In preparation for configuring a new SSO connection, you need to know which adapter instance to use (see “Configuring Adapter Mapping and User Lookup” on page 370). (No adapters are required for a connection that uses only the Attribute Query profile—see “Configuring the Attribute Query Option” on page 401.)

If you intend to pass attribute values to an adapter instance from a local data store, you must define the data store during this configuration, if you have not done so already (see “Managing Data Stores” on page 116).
Chapter 7
Service Provider SSO Configuration

Accessing IdP Connections

You can create or modify connections directly via the Main Menu. Note that the menu displays the four most-recently modified connections. To view a list of all IdP connections, click the Manage All IdP link.

From the Main Menu

From the Main Menu, you can configure a new connection, modify an existing connection, or view connections.

Tip: To copy or delete connections or to find connection drafts, click Manage All IdP (see “From the Manage Connections Screen” on page 355).

Note that long connection names are truncated for this display and the list is limited to four connections, chronologically ordered according to most recently edited. The full connection names and a complete list are displayed on the Manage Connections screen (see “From the Manage Connections Screen” on page 355).

To begin configuring a new connection:

- Click Create New under IdP Connections on the Main Menu.

Tip: The fastest way to create a new connection is to click Manage All IdP and copy the connection with similar settings (see “From the Manage Connections Screen” on page 355).

To modify a connection:

1. Click the connection name under IdP Connections on the Main Menu. Only the four most recently edited connections are displayed. To see all connections, including drafts, click Manage All IdP.
2. On the Activation & Summary screen, click the heading for the information you want to change.
3. Make your change and click **Save**.

**Note:** If **Save** is not available, it means your modification requires other changes or you are editing a screen that is part of a series of subtasks. Click **Next** and continue making indicated changes. The **Done** button indicates that further changes in the task are optional. When you have no further changes, click **Done** and then click **Save** on the task summary screen.

### From the Manage Connections Screen

From the Manage Connections screen you can:

- Create a new connection.
- Modify or copy an existing connection.
- Continue working on a connection draft.
- Delete a connection—if it is not active or referenced in other parts of the configuration (In Use).
- Export individual connection configurations.

**Note:** The connection export function results in an XML file that you can modify and import into another PingTrust server acting in the same federation role (IdP or SP) at your site (see “Connection Management Service” on page 554). You can also automate this process (see “Automating Configuration Migration” on page 78).

- Export metadata about a connection to expedite your partner's corresponding configuration (see “Exporting Metadata” on page 57).

On this screen you can also globally override transaction logging levels set for individual connections or restore connection-based logging (see “Runtime Transaction Logging” on page 47).

**Tip:** A check-mark icon next to a Connection Name indicates that the connection has been checked for configuration errors. For more information about connection-validation features associated with this screen, see “Managing IdP Connection Validation” on page 357.
To access the Manage Connections screen:

▶ Click Manage All IdP under IDP Connections on the Main Menu.

To begin configuring a new connection:

▶ Click Create Connection on the Manage Connections screen.

See “Managing IdP Connections” on page 353 for step-by-step information.

To copy a connection:

1. Click Copy under Action for the connection you want to copy.
2. Update the Connection ID (of the partner) and the Connection Name in the General Info screen (see “General Connection Information” on page 360).
3. Make any further changes needed for the new connection.

To edit a connection or continue working on a draft:

▶ Click the Connection Name link.

For a draft, you will return to where you left off.

To export a connection:

1. Click Export Connection under Action for the connection.
2. Save the XML file on your file system.

You can change the name of the file, but keep the XML extension.

Tip: You can import the connection programmatically or manually into another instance of PingFederate acting in the same role (see “Connection Management Service” on page 554).

To export connection metadata:

1. Click Export Metadata under Action for the connection.

This action takes you to the Export Metadata screen flow, with the connection selection preset (see “Exporting Metadata” on page 57).
2. Complete the steps remaining in the Export Metadata screen flow (starting at Step 4 under “To export connection metadata:” on page 58).
To delete a connection:
1. Under Action, click Delete for the connection.
   (To undo the deletion, click Undelete.)

   Note: The Delete function is not available if the connection is Active or In Use.

2. To confirm the deletion, click Save.

To sort the list of connections:
   ▶ Click the arrow next to any column heading to sort the list based on that column.

To filter the list by Virtual ID, Protocol and/or Status:
   ▶ Select a filter criterion from either or both of the drop-down lists.

   Note: The Virtual ID displays the default virtual server ID of the connection, if any (see “General Connection Information” on page 360).

To override connection-based transaction logging:
1. Select On under Logging Mode Override.
2. Choose the logging mode you want to use for all connections.

To restore connection-based transaction logging:
   ▶ Select Off under Logging Mode Override.

Managing IdP Connection Validation

By default PingFederate automatically validates all existing connections before displaying the Manage Connections screen. This validation ensures that any updates to supporting components—adapters or data-store configurations, for example—have not invalidated any connection settings.

If such errors are found, a warning icon appears next to the Connection Name.

To correct errors:
   ▶ Click the Connection Name to reach the top-level task in which reconfiguration is needed, and to see the error message. Then navigate into deeper tasks using Configure . . . buttons to find a link to the screen that needs updating.

   (For more information about console navigation, see “About Tasks and Steps” in the “Console Navigation” chapter of Getting Started.)

   Note that the connection validation time increases with the number of connections and when connections are configured to access data stores for adapter mapping. Consequently, there may be noticeable delays in displaying the Manage Connections screen. For this reason, PingFederate provides a way to turn off the automatic validation under Server Settings (see “Setting System Options” on page 110).

When validation is turned off, administrators can check connections manually on the Manage Connections screen. A question-mark icon indicates the connection has not been validated. In addition, Action links are disabled (except for Delete, if the connection is Inactive and/or In Use). You may, however, still edit the connection by clicking its name.
When automatic validation is disabled, use one of the following procedures to validate connections:

**To validate a single connection:**

- Click icon next to the Connection Name.

**Note:** Validating a single connection does not check connectivity for any configured data-store lookups (see “Selecting an Adapter Data Store” on page 374). However, this check is performed when you access the connection for editing.

**To validate all connections (including for data-store connectivity):**

1. Click **Check All Connections for Errors**.
2. When prompted, click **Yes**.

### Choosing an IdP Connection Type

Indicate on the Connection Type screen whether the connection to this partner is for Browser SSO, WS-Trust STS, OAuth SAML, and/or Inbound Provisioning (see “Connection Types” on page 5).

**Note:** You can add STS, OAuth, and Inbound Provisioning support to any existing SSO connection, or vice versa, at any time.

If your federation deployment supports multiple protocols, then for new SSO connections you can also select the applicable protocol on the Connection Type screen (see “Choosing Roles and Protocols” on page 106).

**Note:** If your partner’s deployment also supports multiple protocols and you intend to communicate using more than one, then you must set up a separate connection for each protocol.

- To configure a connection for secure browser-based SSO, select **Browser SSO Profiles** and the **Protocol** (if necessary).
- To configure a connection for WS-Trust STS, make that selection.
- To configure an OAuth SAML Grant connection, make that selection.

**Note:** This option is available only after you configure an access token plug-in (see “Access Token Management” on page 165).
To configure an Inbound Provisioning connection, make that selection.

Note: The Inbound Provisioning option is active only after you enable the Inbound Provisioning protocol on the Roles & Protocol screen (see “Choosing Roles and Protocols” on page 106).

(Optional) If your PingFederate license manages connections by groups, then you can select a group for this connection. This option is not displayed for unrestricted or other types of licenses.

Choosing IdP Connection Options

On the Connection Options screen, you can choose to enable JIT Provisioning in conjunction with Browser SSO (see “Just-in-Time Provisioning” on page 35).

You can also choose to map user attributes for persistent grants used by the optional PingFederate OAuth Authorization Server (see “About OAuth” on page 10).

Note: The OAuth Attribute Mapping option is active only when the OAuth 2.0 Authorization Server (AS) role is enabled on the Roles & Protocol screen (see “Choosing Roles and Protocols” on page 106).

For SAML 2.0, you also have the option of configuring the Attribute Query profile, with or without SSO (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

Note: This screen is presented only for browser-based SSO connections (see “Choosing an IdP Connection Type” on page 358).

To create a connection for browser-based SSO, ensure that Browser SSO is selected and click Next.

Importing IdP Metadata

If you are using one of the SAML protocols and have received a metadata file from your partner, click Browse on the Import Metadata screen, select the file, and click Next.

Note: If the endpoints in the metadata file share the same base URL (protocol, hostname, and port), PingTrust uses this information to populate the Base URL field (see “General Connection Information” on page 360). Consequently, individual endpoints on other screens do not include this information—only relative paths are shown.
Chapter 7
Service Provider SSO Configuration

If you are not using a metadata file, click Next on the Import Metadata screen.

Importing a Certificate

The Import Certificate screen appears only if the metadata file you have chosen to import is signed and the certificate needed to verify the signature is not contained in the file.

> Click Browse to locate and open the signature verification certificate for this partner.

Viewing the Summary

The Metadata Summary screen provides security information about an imported metadata file, including whether the file was signed and, if so, the trust status of the certificate used to verify the signature.

General Connection Information

On the General Info screen, you provide a required unique identifier and display name for a connection, as well as optional contact information. In addition, on this screen you can define a default error message that end users will see in the event that SSO fails, and you can set the level of transaction logging for this connection partner (see “Runtime Transaction Logging” on page 47).
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partner’s Entity ID/Issuer/Partner’s Realm (Connection ID)</td>
<td>(Required) The published, protocol-dependent, unique identifier of your partner. For a SAML 2.0 connection, this is your partner’s SAML Entity ID. For a SAML 1.x connection, this is the Issuer your partner advertises. For a WS-Federation connection, this is your partner’s Realm. This ID may have been obtained out-of-band or via a metadata file if you are using a SAML protocol (see ). For STS-only connections, this ID can be any unique identifier.</td>
</tr>
<tr>
<td>Connection Name</td>
<td>(Required) A plain-language identifier for the connection—for example, a company or department name. This name is displayed in the connection list on the Main Menu.</td>
</tr>
<tr>
<td>Virtual Server IDs</td>
<td>If you want to identify your server to this connection partner using an ID other than the one you specified under Server Settings (see ), enter a virtual server ID in this field and click Add. Enter additional virtual server IDs as needed (see ).</td>
</tr>
</tbody>
</table>
## Chapter 7
### Service Provider SSO Configuration

**For a new connection:**

- Fill in the information needed and click **Next**.

  **Connection ID** and **Connection Name** are required (see “Field Descriptions” above).

  Note that the values in **Virtual Server IDs** identify your own federation deployment for this connection only and override the ID you specified under **Server Settings** (see “Federation Server Identification” on page 37).

**For an existing connection:**

- If you are editing existing information, modify the fields as needed and click **Save**.

### Configuring Browser SSO

Browser-based SSO (also, Browser SSO) is another term for secure SSO, which relies on a user’s Web browser and HTTP to broker XML identity-federation messaging between an IdP and an SP (in contrast to WS-Trust STS messaging, which is typically application-driven across the back channel and does not require browser mediation).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base URL</td>
<td>The fully qualified hostname and port on which your partner’s federation deployment runs (e.g., <code>https://www.pingidentity.com:9031</code>). This entry is an optional convenience, allowing you to enter relative paths to specific endpoints, instead of full URLs, during the configuration process.</td>
</tr>
<tr>
<td>Company</td>
<td>The name of the partner company to which you are connecting.</td>
</tr>
<tr>
<td>Contact Name</td>
<td>The contact person at the partner company.</td>
</tr>
<tr>
<td>Contact Number</td>
<td>The phone number of the contact person at the partner company.</td>
</tr>
<tr>
<td>Contact Email</td>
<td>The email address for the contact person at the partner company.</td>
</tr>
<tr>
<td>Error Message</td>
<td>If an error occurs on this server, the end user’s browser may be redirected (in a default situation) to an error page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87). The default entry in this field is used to localize the message. For information about how to find and change the default English message and how use the PingFederate localization feature, see “Localization” on page 92. If localization is not needed, you may also specify a message directly in this field to change the default.</td>
</tr>
<tr>
<td>Logging Mode</td>
<td>The level of transaction logging applicable for this connection (see ).</td>
</tr>
</tbody>
</table>
To continue, click **Configure Browser SSO**.

### Connection Configuration Steps

Many steps involved in setting up a federation connection are protocol-independent; that is, they are required steps for all connections, regardless of the associated standards (see the “Supported Standards” chapter in *Getting Started*). Also, for any given connection, some configuration steps are required under the applicable protocol, while others are optional. Still others are required only based on certain selections. The PingFederate administrative console determines the required and optional steps based on the protocol and dynamically presents additional requirements or options based on selections.

The following sections provide sequential information about every step you might encounter while configuring browser-based SSO, regardless of the protocol you are using for a particular connection.

**Note:** The configuration screens represented in this chapter show “SAML 2.0” in their left corners, unless they are exclusive to WS-Federation or SAML 1.x setup requirements. When the SAML 2.0 screens are also applicable to SAML 1.x and/or WS-Federation connections, the SAML 2.0 representations and discussions also apply to the other protocols, unless otherwise indicated.

After configuring SSO settings, you will normally need to configure authentication credentials, the range of which depends on your SSO selections (see “Configuring Security Credentials” on page 424). Also, other configuration tasks may remain to be configured for new or modified connections, depending on selected connection options (see “Choosing IdP Connection Options” on page 359).

**Important:** For new connections you must completely configure these SSO settings and subsequent tasks before you can save the connection on the Activation & Summary screen. Until then, the configuration is temporary and can be lost; the console times out after several minutes of inactivity. At any time, however, you can click **Save Draft**, which is available on most screens after you enter General Information (see “Console Buttons” in the “Console Navigation” chapter of *Getting Started*).

Use the lists and links (or page references) below to find specific information about steps that may apply to your SSO connection requirements:

**SAML 2.0 SSO Steps**

- “Choosing SAML Profiles” on page 364
- “User-Session Creation” on page 367
– “Selecting an Identity-Mapping Method” on page 367
– “Defining an Attribute Contract” on page 369
– “Configuring Adapter Mapping and User Lookup” on page 370

• “Configuring SAML Protocol Settings” on page 392
  – “Specifying SSO Service URLs (SAML)” on page 393
  – “Specifying SLO Service URLs” on page 395
  – “Choosing Allowable SAML Bindings (SAML)” on page 396
  – “Setting an Artifact Lifetime (SAML 2.0)” on page 396
  – “Specifying Artifact Resolver Locations” on page 397
  – “Configuring Signature Policy” on page 398
  – “Configuring XML Encryption Policy (for SAML 2.0)” on page 399

WS-Federation SSO Steps
• “User-Session Creation” on page 367
  – “Selecting an Identity-Mapping Method” on page 367
  – “Defining an Attribute Contract” on page 369
  – “Configuring Adapter Mapping and User Lookup” on page 370

• “Configuring SAML Protocol Settings” on page 392
  – “Specifying a Service URL (WS-Federation)” on page 394

SAML 1.x SSO Steps
• “Choosing SAML Profiles” on page 364
• “User-Session Creation” on page 367
  – “Selecting an Identity-Mapping Method” on page 367
  – “Defining an Attribute Contract” on page 369
  – “Configuring Adapter Mapping and User Lookup” on page 370

• “Configuring SAML Protocol Settings” on page 392
  – “Specifying SSO Service URLs (SAML)” on page 393
  – “Choosing Allowable SAML Bindings (SAML)” on page 396
  – “Specifying Artifact Resolver Locations” on page 397
  – “Configuring Signature Policy” on page 398

Choosing SAML Profiles

A SAML profile is the message-interchange scenario that you and your federation partner have agreed to use (see “Federation Planning Checklist” on page 36). For SAML 2.0, PingFederate supports all IdP- and SP-initiated SSO and SLO profiles.

The SAML 1.x implementation supports standard IdP-initiated SSO as well as nonstandard SP-initiated SSO.

For information on typical SSO/SLO profile configurations, including illustrations, see the “Profiles” sections in the “Supported Standards” chapter in Getting Started.
You can configure profiles individually or all together. PingFederate presents the correct configuration steps to fit your choices. Steps that apply to one SSO or SLO profile often apply to others and are reused automatically across profiles.

**Note:** For SAML 1.x, IdP-initiated SSO is assumed and the specifications do not support SLO; the only choice on this screen is SP-initiated SSO (see “SAML 1.x Profiles” in the “Supported Standards” chapter of *Getting Started*).

For WS-Federation, the SAML Profiles screen is not presented.

**To reach this screen:**
1. Click a connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the IdP Connection tab.
3. Click **SAML Profiles** on the Summary screen.

**To configure profiles:**
1. Select the profile(s) applicable to this connection and click **Next**.
   - For SAML 2.0 connections, you must select an SSO profile before you can enable SLO.
2. Continue through the remaining connection-configuration tasks.

**About IdP-Initiated SSO**

When PingFederate is operating as an SP, the IdP-initiated SSO profile configuration defines the message-transport mechanisms (bindings) your enterprise has agreed to allow for receiving SAML assertions, plus any digital signature verification requirements for inbound assertions (see “Security Infrastructure” on page 25).

For illustrations of typical profile and binding scenarios, see the “Profiles” sections in the “Supported Standards” chapter in *Getting Started*.

For this configuration you need to know:
- The transport binding(s) to which you and your partner have agreed
- The certificate to be used for verifying incoming digital signatures from your IdP
  (optional for the artifact binding)

When Artifact is an allowable inbound SAML binding, you also need to know the endpoint(s) to your partner's Artifact Resolution Service(s) and the SOAP client authentication mechanism to use: either HTTP Basic, SSL client certificates, a digital signature, or a combination of these mechanisms.
## About SP-Initiated SSO

The SP-initiated SSO profile configuration defines the message-transport mechanisms (bindings) and security requirements for sending authentication requests and receiving assertions when your site initiates SSO transactions.

For SAML 1.x, the SP-initiated SSO profile is also known as the “destination-first” profile, which was added as a supported “non-normative” use case after the release of the SAML 2.0 specifications.

For illustrations of typical profile and binding scenarios, see the “Profiles” sections in the “Supported Standards” chapter in Getting Started.

For this configuration you will need to know:

- The endpoint URL(s) for your IdP’s Single Sign-on Service(s)
- The transport bindings to which you and your partner have agreed (inbound and outbound)
- The certificates you will use to sign outbound authentication requests and to verify incoming digital signatures from your IdP

When Artifact is an allowable inbound SAML binding, you also need to know the endpoint(s) to your partner’s Artifact Resolution Service(s) and the SOAP client authentication mechanism to use: either HTTP Basic, SSL client certificates, a digital signature, or a combination of these mechanisms.

## About IdP-Initiated SLO

The SAML 2.0 IdP-initiated SLO profile configuration defines the message-transport mechanisms (bindings) and security requirements that you and your partner have agreed Upon for exchanging SLO requests and responses.

**Note:** SLO is not supported by the SAML 1.x specifications.

For more information about SLO, see “Single Logout” in the “Supported Standards” chapter of Getting Started.

For this configuration you need to know:

- The transport bindings that you and your partner have agreed upon to send SLO requests and receive responses
- The certificates to be used for signing outgoing messages and for verifying incoming digital signatures from your IdP (optional for the artifact binding)
- The URL(s) of your IdP’s Single Logout Service(s)
- The URL of your IdP’s Artifact Resolution Service(s) (to resolve artifacts from the IdP) and SOAP client authentication requirements

## About SP-Initiated SLO

The SAML 2.0 SP-initiated profile configuration for SLO defines the message-transport mechanisms (bindings) and security requirements that you and your partner have agreed upon for exchanging SAML requests and responses.

**Note:** SLO is not supported by the SAML 1.x specifications.
Managing IdP Connections

For more information about SLO, see “Single Logout” in the “Supported Standards” chapter of Getting Started.

For this configuration you need to know:

- The transport bindings that you and your partner have agreed upon to send SLO requests and receive responses
- The certificates to be used for signing outgoing messages and for verifying incoming digital signatures from your IdP (optional for the artifact binding)
- The URL(s) of your IdP’s Single Logout Service(s)
- The URL of your IdP’s Artifact Resolution Service(s) (to resolve artifacts from the IdP) and SOAP client authentication requirements

User-Session Creation

As an SP, you must specify how you use information sent from the IdP in SSO assertions to create user sessions for enabling access to protected resources at your site. This configuration includes:

- Choosing an identity-mapping method (see “Selecting an Identity-Mapping Method” next).
- Defining the attribute contract you will use with this partner, if any (see “Defining an Attribute Contract” on page 369).
- Configuring instances of one or more adapters (see “Configuring Adapter Mapping and User Lookup” on page 370) and specifying how they are used to fulfill the adapter contract.

To continue, click Configure User-Session Creation.

Selecting an Identity-Mapping Method

PingFederate allows an SP to use either account linking or account mapping to associate remote users with local accounts for SSO between business partners (see “Identity Mapping” on page 17). At the Identity Mapping step, you choose which method to use with a particular IdP connection. You and your partner may want to decide in advance which option to use (see “Federation Planning Checklist” on page 36).
If your site is using account linking, then establishing an **attribute contract** is not required. Depending on your partner agreement, however, you may choose to supplement the account link with an attribute contract. In this configuration the account link is used to determine the user's identity, while the additional attributes might be used for authorization decisions, customized Web pages, and so on, at your site (see “About Attributes” on page 19).

---

**Important:** If you have previously set up a configuration to use an attribute contract and want to change the configuration to use account linking without additional attributes, then the existing attribute contract will be discarded.

---

Account linking can be used with either a clear, standard name identifier or an opaque pseudonym.

> If you want to dynamically associate remote users with local accounts using a known attribute to identify a user—for example, a username or email address—then select **Account Mapping**.

Account mapping uses the value passed in the SAML assertion's `SAML_SUBJECT` and associated user attributes to create an association between a remote user and a local account.

**Tip:** If you are using PingFederate’s JIT Provisioning, choose Account Mapping (see “Using Just-in-Time Provisioning” on page 403).

> If you want to create a long-term association between a remote user and a local account, then select **Account Linking** on the Identity Mapping screen.

To set up an attribute contract to use in conjunction with account linking, select the checkbox next to “The assertion includes attributes . . .” after selecting **Account Linking**.

**Tip:** An SP PingFederate uses a default, Hypersonic database to handle account linking. You can use your own database instead, as needed. For more information, see “Configuring an LDAP Connection” on page 121.

---

**To reach this screen:**

1. Click the connection name on the Main Menu.
   Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the IdP Connection tab.
3. Click **Configure Browser SSO**.
Managing IdP Connections

4. Click **User-Session Creation** under the Browser SSO tab.
5. Click **Configure User-Session Creation**
6. Click **Identity Mapping** on the Summary screen.

**Defining an Attribute Contract**

An attribute contract is the set of user attributes that you and your partner have agreed will be sent in SAML assertions for this connection (see “Attribute Contracts” on page 19). You identify these attributes on this screen.

**SAML_SUBJECT** is always sent in a SAML assertion and contains the name identifier of the user requesting SSO. When you select **account mapping** as the identity mapping technique, the **SAML_SUBJECT** is available to help map the incoming user to a local ID on your system (see “Selecting an Identity-Mapping Method” on page 367).

For **account linking**, the **SAML_SUBJECT** contains an identifier that the SP server uses to make a permanent association between the remote user and a local account. The **SAML_SUBJECT** itself is not available to the SP application and thus does not appear in the Attribute Contract on this screen.

Optionally, you can mask the values of attributes (other than **SAML_SUBJECT**) in the log files that PingFederate writes when it receives assertions (see “Attribute Masking” on page 23).

To add an attribute:

1. Enter the attribute name in the text box. Attribute names are case-sensitive and must correspond to the attribute names expected by your partner.
2. (Optional) Select the checkbox under Mask Values in Log.
3. Click **Add**.

To modify an attribute name or masking selection:
1. Click **Edit** under Action for the attribute.
2. Make the change and click **Update**.

---

**Note:** If you change your mind, ensure that you click the **Cancel** link in the Actions column, not the **Cancel** button, which discards any other changes you might have made in the configuration steps.

To delete an attribute:
- Click **Delete** under Action for the attribute.

---

### Configuring Adapter Mapping and User Lookup

Remote users arriving at your site via an SSO request do so in order to use specific applications or gain access to protected resources. Based on the nature of the business relationship and the agreement with your partner, you may be expected to provide access to these applications. Therefore, integration between your federation SP server and local applications is important.

The PingFederate server for an SP uses integration adapters to identify the user to your applications based on attributes sent in an assertion. The server uses this information to create a local session that enables access to user-requested resources (the “target”) at your site. (See “SSO Integration Kits and Adapters” on page 14.)

Each adapter instance requires a set of attributes into which you map values found in the assertion. You can map additional attributes, as needed, from local data stores, or you can use static or variable text values. An adapter instance will fail to create a local session for the incoming user if it is unable to find values for each of its required attributes.

You must associate at least one adapter instance with an IdP connection. If you have multiple integration requirements—for example, if you are using more than one IdM system or an application not covered by a centralized system—then you should map multiple adapter instances.

When adapters are restricted to certain virtual server IDs, the allowed IDs are displayed under the Virtual Server IDs column.

---

**Note:** If you configure only one adapter instance for a connection, the server will use that instance at runtime without checking for any associated URLs (see “Mapping URLs to Adapter Instances” on page 342).
Managing IdP Connections

To reach this screen:
1. Click the connection name on the Main Menu.
   - Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation

To begin mapping an adapter:
- Click Map New Adapter Instance and follow the configuration screens (see the following sections for more information).

To begin modifying an existing adapter mapping:
- Click the Adapter Instance Name and navigate through the steps to the information you want to change.

Choosing an Adapter Instance

An SP adapter instance is available for use within an IdP connection only after it has been deployed in PingFederate.
Chapter 7
Service Provider SSO Configuration

To select an adapter instance:

- Choose an adapter instance from the drop-down menu and click Next to continue.

(Optional) To override any SP adapter instance, select the Override Instance Settings checkbox (see “Overriding SP Adapter Instances” on page 372).

If the adapter instance you need is not available, click Manage Adapter Instances to define one or more adapter instances you need for this connection.

Note that an adapter instance can be mapped only once per connection.

Tip: Adapter contracts for some adapters can be customized for individual connection requirements (see “Configuring SP Adapters” on page 338).

Overriding SP Adapter Instances

Overrides at the connection level simplify adapter management by allowing you to create a small set of adapter instances that define the base configuration requirements and then overriding settings at the connection level as needed.

You may override any SP adapter settings at the connection level either during or after connection mapping.

Alternatively, you can override any adapter instance to apply to several connections, see “Hierarchical Plug-in Configurations” on page 17 for more information about adapter overrides.

Note: Any changes to the base adapter instance are propagated to a connection provided the same changes are not overridden for the connection.
Click **Override Instance Settings**.

On each of the settings screens, make your changes, and then click **Next**. When you are finished, click **Done** to continue with SP adapter mapping.

**Note:** Override adapter-setting screens are functionally identical to those used for creating a new adapter connection. Refer to the table below to find sections in this manual containing configuration information and procedures.

The display of some screens listed in the table depends on the type of adapter you are configuring.

For information about the override adapter-setting screens, depending on the type of setting, use the following table:

<table>
<thead>
<tr>
<th>Override Screen</th>
<th>Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter</td>
<td>See “Configuring an Adapter Instance” on page 340.</td>
</tr>
<tr>
<td>Actions</td>
<td>See “Choosing Adapter Actions” on page 341.</td>
</tr>
<tr>
<td>Extended Contract</td>
<td>See “Extending Adapter Contracts” on page 341.</td>
</tr>
</tbody>
</table>

To remove any adapter connection override, clear the **Override Instance Settings** checkbox, and then complete the rest of the setup.

**To reach this screen:**
1. Click the connection name on the Main Menu.
2. Click **Manage All IdP**, if needed, to see a full list of connections.
3. Click **Browser SSO** under the IdP Connection tab.
4. Click **Configure Browser SSO**.
5. Click **User-Session Creation** under the Browser SSO tab.
6. Click **Configure User-Session Creation**.
7. Click **Adapter Instance** on the Summary screen.
8. Click **Next**.

**Restricting an SP Adapter to certain virtual Server IDs**

When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), you have the option to enforce authentication requirements by restricting an adapter to certain virtual server IDs. This optional setting can be applied to each adapter added to the connection using virtual server IDs. By default, no restriction is imposed.
Chapter 7
Service Provider SSO Configuration

To restrict an adapter to a subset of the available virtual server IDs:
1. Select the **Restrict Virtual Server IDs** checkbox.
2. In the **Allowed Virtual Server IDs** area, select one or more virtual server IDs that you want to allow for this SP adapter.
3. Click **Next**.

**To reach this screen:**
1. Click the connection name on the Main Menu.
   Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the IdP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **User-Session Creation** under the Browser SSO tab.
5. Click **Configure User-Session Creation**
6. Click **Adapter Mapping & User Lookup** on the Summary screen.
7. Click the Adapter Instance Name.
8. Click **Virtual Server IDs** on the Summary screen.

**Note:** The Virtual Server IDs screen is only available for connections using at least one virtual server ID, see “Federation Server Identification” on page 37.

**Selecting an Adapter Data Store**

To populate the attributes required by the adapter (the adapter contract), you can use values supplied by SAML assertions from the IdP exclusively, or in addition to values retrieved from local user-data stores (see “Managing Data Stores” on page 116).

![Table of Adapter Data Store](image)

- If you choose to look up additional values, click the applicable button and then **Next**. This selection allows you to identify data sources and specify lookup queries in subsequent screens (see “Data Store Setup” next).

  Or:

  If you choose not to look up additional values, click the applicable button (if it is not already selected) and then **Next**. This selection takes you directly to a
screen where you can map attribute values from the assertion (see “Configuring Adapter Contract Fulfillment” on page 383).

**Tip:** To determine whether you need to look up additional values, compare your attribute contract against your adapter contract (see “Defining an Attribute Contract” on page 369 and “Choosing an Adapter Instance” on page 371). If the adapter requires more information, determine whether your local data stores can supply it. (You can also choose to use text constants or expressions for certain information—see “Configuring Adapter Contract Fulfillment” on page 383.)

### Data Store Setup

For data-store setup information, refer to the sections indicated in the following steps.

**Note:** As you make selections on configuration screens, ensure that you allow enough time for PingFederate to access your data store and populate drop-down lists.

1. See “Choosing a Data Store” on page 375.
2. See the following sections in this manual, depending on the type of data store:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Related Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>JDBC</td>
<td>• “Selecting a Database Table and Columns” on page 376</td>
</tr>
<tr>
<td></td>
<td>• “Configuring a Database Filter” on page 378</td>
</tr>
<tr>
<td>LDAP</td>
<td>• “Configuring a Directory Search” on page 380</td>
</tr>
<tr>
<td></td>
<td>• “Specifying an LDAP Filter” on page 381</td>
</tr>
<tr>
<td>Custom</td>
<td>• “Configuring Custom Data-Source Filters” on page 383</td>
</tr>
<tr>
<td></td>
<td>• “Selecting Custom Data-Source Fields” on page 383</td>
</tr>
</tbody>
</table>


### Choosing a Data Store

This screen allows you to choose a data store from a previously configured list (see “Managing Data Stores” on page 116). Attribute values extracted from this data store are used in combination with the values from the attribute contract to fulfill the adapter contract for this adapter instance (see “Configuring Adapter Mapping and User Lookup” on page 370).
Chapter 7
Service Provider SSO Configuration

To reach this screen:

1. Click the connection name on the Main Menu.
   - Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation.
6. Click Adapter Mapping & User Lookup under the User-Session Creation tab.
7. Click the Adapter Instance Name.
8. Click Data Store on the Summary screen.
    - If this step is not present, then the use of a data store has not been selected (see “Selecting a Database Table and Columns” on page 376).

To define an attribute source:

- Choose an Active Data Store and click Next.
  - A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click Manage Data Stores to add it (see “Managing Data Stores” on page 116).

Selecting a Database Table and Columns

When you choose to use a database source for attributes, you follow this path through the configuration steps.
On this screen you specify a database table and columns where user data is located. You will specify the user lookup query next (see “Configuring a Database Filter” on page 378).

**Important: (For MySQL users)** To allow for table and column names that may contain spaces, PingFederate inserts double quotes around the names at runtime. To avoid SQL syntax errors resulting from the quotes, add the property `ANSI_QUOTES` to `sql-mode` in the configuration file `my.cnf` (on Unix/Linux) or `my.ini` (on Windows). For example:

```sql
sql-mode="...,ANSI_QUOTES"
```

For more information, see:
- [dev.mysql.com/doc/refman/5.0/en/identifiers.html](http://dev.mysql.com/doc/refman/5.0/en/identifiers.html)

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Schema</strong></td>
<td>Lists the table structure that stores information within a database. Some databases, such as Oracle, require selection of a specific schema for a JDBC query. Other databases, such as MySQL, do not require selection of a schema.</td>
</tr>
<tr>
<td><strong>Table</strong></td>
<td>The name of the table contained in the database. The name is used to construct the SQL query to retrieve data from the data store.</td>
</tr>
<tr>
<td><strong>Add Attribute</strong></td>
<td>Adds the column to be executed in the SQL query to the data store to retrieve the attribute value.</td>
</tr>
<tr>
<td>(Drop-down menu)</td>
<td>The available columns of attribute names for the selected table are displayed. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.</td>
</tr>
</tbody>
</table>
Chapter 7
Service Provider SSO Configuration

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation
6. Click Adapter Mapping & User Lookup under the User-Session Creation tab.
7. Click the Adapter Instance Name.
8. Click Database Table and Columns on the Summary screen.
   If this step is not shown, this connection is not yet configured to use a database
to look up attributes. For information about changing this configuration, see
“Choosing a Data Store” on page 375.

To select a database table and columns for queries:
1. Choose a Schema file (when applicable) from the drop-down list.
2. Choose a Table from the drop-down list.
3. Choose a name under Columns to Return from Select and click Add Attribute.

Tip: Click Refresh if you are updating an existing configuration and
changes may have been made to the database.

Repeat this step for other columns as needed.

Note: You do not need to add a column here for it to be used as
part of a search filter (see “Configuring a Database Filter” next).
Add only attributes from which you need actual values to pass to
the adapter.

Tip: To determine which attributes to look up during a query, click
Adapter Contract to Fulfill to see what information must be collected
(see “Choosing an Adapter Instance” on page 371). Then determine
what information is coming in from the assertion (see “Defining an
Attribute Contract” on page 369). Information not contained in the
Attribute Contract may be pulled from the data store look-up query.

Configuring a Database Filter
On this screen you begin to specify exactly where additional data can be found to
complete the attribute contract when you receive an assertion from this IdP (see
“Creating an Attribute Contract” on page 264).
The JDBC WHERE clause queries your data store to locate a user record. Once the record is located, the configured SELECT statements retrieve the attribute values.

The clause is in the form:

```
WHERE column1=value1 [AND column2=value2] [OR ...]
```

The left side of the first variable pair uses a column name in the database table you selected (see “Selecting a Database Table and Columns” on page 376).

The right side generally uses values passed in from the assertion. Possible variables for these, including the correct syntax, are listed under Assertion Values.

You can also apply additional search criteria from your own database, using any other columns from the targeted table.

**Tip:** Click “View List of Columns . . .” to see a list from which to copy and paste.

For more information about WHERE clauses, consult your DBMS documentation.

**Example:**

```
userid='${username}'
```

In this example `userid` is the name of a column in the JDBC data store. On the right side, `'{username}'` returns the value of the `username` from the assertion.

**Important:** You must use the `{}` syntax to retrieve the value of the enclosed variable and use single quotation marks around the `{}` characters.

**To reach this screen:**

1. Click the connection name on the Main Menu. Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click **Configure User-Session Creation**

6. Click **Adapter Mapping & User Lookup** under the User-Session Creation tab.

7. Click the Adapter Instance Name.

8. Click **Database Filter** on the Summary screen.

   If this step is not shown, this connection is not yet configured to use a database to look up attributes. For information about changing this configuration, see “Choosing a Data Store” on page 375.

**To construct the WHERE clause:**

1. Enter the statement in the space provided, following the guidelines and example above.
   The initial WHERE is optional.

2. Ensure the syntax and variable names are correct.

   When you click **Next**, you will map attribute values returned from the database into the attribute contract (see “Selecting an Adapter Data Store” on page 374).

### Configuring a Directory Search

When you choose to use an LDAP source for attributes, you follow this path through the configuration steps.

On this screen you begin to specify exactly where additional data can be found to supply to the SP adapter in order to access a resource on your system (see “SSO Integration Kits and Adapters” on page 14).

#### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. This field is optional if records are located at the LDAP root.</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Determines the node depth of the query. Select Subtree, One level or Object.</td>
</tr>
</tbody>
</table>
Managing IdP Connections

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation
6. Click Adapter Mapping & User Lookup under the User-Session Creation tab.
7. Click the Adapter Instance Name.
   If this step is not shown, this connection is not yet configured to use LDAP to look up attributes (see “Choosing a Data Store” on page 375).

To select LDAP attributes:
1. (Optional) Enter a Base DN.
2. Select a Search Scope.
3. Select a Root Object Class.
4. Under Attributes to return from search, choose an attribute and click Add Attribute.
   Note that the attribute Subject DN is always returned by default.
5. Repeat the last step for other attributes as needed.
6. (Optional) Change the Search Scope or the Root Object Class if you want attributes from other locations.

Note: You do not need to add an attribute here for it to be used in a search filter (see “Specifying an LDAP Filter”). Add only attributes from which you need values to map to the adapter.

Specifying an LDAP Filter

The LDAP filter queries the data store to retrieve a user record associated with a particular value (or values) from the assertion. The filter is in the form:

(attribute=${value})

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root Object Class</td>
<td>Specifies the object type within the LDAP hierarchy from which attributes will be returned.</td>
</tr>
<tr>
<td>Attribute</td>
<td>The available attribute names for the selected directory structure. Select the names associated with the attributes that you would like to return from the query.</td>
</tr>
</tbody>
</table>
The left-side variable is an attribute from the data store (see “Configuring a Directory Search” on page 380).

The right side generally uses values passed in from the assertion.

You can also apply additional search criteria from your data store, using any other attributes from the targeted object classes.

**Tip:** Click “View List of Available LDAP Attributes” for a list from which you can copy and paste.

For general information about search filters, consult your LDAP documentation.

**Example:**

```lang
(UNAME=${username})
```

In this example UNAME is the name of an attribute in the LDAP data store. On the right side, ${username} returns the value of username in the assertion.

**Important:** You must use the ${} syntax to retrieve the value of the enclosed variable.

### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>A filter narrows a search to locate requested data by either including or excluding specific records. An LDAP filter includes the attributes in the search and the value or range of values that the search is attempting to match. Searches are conducted by using at least three components: 1) at least one attribute (attribute data type) to search on, 2) a search filter operator that will determine what to match, and 3) the value of the attribute being sought. Searches must have at least one of each of these three components.</td>
</tr>
</tbody>
</table>
To reach this screen:

1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation
6. Click Adapter Mapping & User Lookup under the User-Session Creation tab.
7. Click the Adapter Instance Name.
8. Click LDAP Filter on the Summary screen.
   If this step is not shown, this connection is not configured to use LDAP to look up attributes (see “Choosing a Data Store” on page 375).

To construct the LDAP filter:

1. Enter the statement in the space provided, following the guidelines and example above.
2. Ensure the syntax and variable names are correct.
3. Click Next.

Configuring Custom Data-Source Filters

When you choose to use a custom source for attributes, you follow this path through the configuration steps.

On this screen you specify a filter, or lookup query, for your custom data source. This screen display and the syntax of the filter depends on your developer's implementation of the custom source SDK.

Selecting Custom Data-Source Fields

On the Configure Custom Source Fields screen, you can choose from among the fields shown to map to the adapter contract. These choices are supplied by the driver implementation. Select only those needed to fulfill the attribute contract for this partner connection.

Configuring Adapter Contract Fulfillment

The last step in configuring an adapter is to map each attribute required for the adapter contract to a value (see “Selecting an Adapter Data Store” on page 374). These are the values that will be used to create a local session. An SSO operation fails if the SP is unable to fulfill the mapping requirements defined here.

Map attributes from one of these Sources:

- Account Link
  This source appears only if you have elected to use account linking (see “Selecting an Identity-Mapping Method” on page 367). When you make this selection, the associated Value drop-down list is populated with Local User ID.
Normally, you would map this identifier to target an adapter attribute that represents the local user ID.

- **Assertion**
  Values are contained in the assertions from this IdP. When you make this selection, the associated `Value` drop-down list is populated by the attribute contract (see “Defining an Attribute Contract” on page 369).

- **Context**
  Values are returned from the context of the transaction at runtime.

**Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **JDBC/LDAP/Custom**
  Values are returned from your query. When you make this selection, the Value list is populated by the database columns or LDAP or custom attributes you identified for this data store (see “Selecting a Database Table and Columns” on page 376, “Configuring a Directory Search” on page 380, or “Selecting Custom Data-Source Fields” on page 383).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the assertion, using the `$ {attribute}` syntax.

  You can also enter values from your data store, when applicable, using this syntax:

  `$ {ds.attribute}`

  where `attribute` is any of the data store attributes you select.

**Tip:** Two other variables are also available: `$ {SAML_SUBJECT}` and `$ {TargetResource}`. `SAML_SUBJECT` is the initiating user (or other entity). `TargetResource` is a reference to the protected application or other resource for which the user requested SSO access; this variable is available only if specified as a query parameter for the relevant PingFederate endpoint (either as `TargetResource` for SAML 2.0 or `TARGET` for SAML 1.x—see “Application Endpoints” on page 529).

There are a variety of reasons why you might hard code a text value. For example, if your Web application provides a consumer service, you might want to supply a particular promotion code for this partner.
Managing IdP Connections

To reach this screen:

1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click User-Session Creation under the Browser SSO tab.
5. Click Configure User-Session Creation
6. Click Adapter Mapping & User Lookup under the User-Session Creation tab.
7. Click the Adapter Instance Name.
8. Click Adapter Contract Fulfillment on the Summary screen.

To map attributes:

1. Choose a Source for each Target attribute (see descriptions of each Source type above).
2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.
3. Click Done.

Identifying Issuance Criteria (Optional)

Use this screen to define criteria PingFederate can evaluate to determine whether the user is authorized to continue the SSO transaction using the SP adapter (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access protected resources.
Chapter 7
Service Provider SSO Configuration

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   
   Associated attributes appear in the Attribute Name drop-down list:
   
   - Assertion – Select to access attributes from the assertion.
   - Context – Select to use values returned from the context of the transaction at runtime.

   **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

   - LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.

   **Note:** PingFederate appends a description in parentheses for data stores of the same type.

   - Mapped Attributes – Select to access the required attributes from the adapter.

2. Select an attribute name.

3. Select the Condition you want to apply.

   **Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

   Error results are handled in one of two ways:

   - **Redirect** – When an InErrorResource URL is provided, the value of the Error Result field is used by an ErrorDetail query parameter in the redirect URL.

   - **Template** – When an InErrorResource URL is not provided, the value of the Error Result field is used by the variable $errorDetail in the sp.sso.error.page.template.html template (for more information on this and other user-facing templates, see “Customizing User-Facing Screens” on page 87).

   **Note:** Using an error code in the Error Result field allows the error template or a Web application to process the error result in a variety of ways—for example, a redirect to another page or e-mail to an administrator.

   If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click **Add**.
7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the **Show Advanced Criteria** button to appear (see “Enabling and Disabling Expressions” on page 573).

**Important:** When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

- Use the in-line editor box to enter the OGNL expression.
  
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.

- Use the **Error Result** box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

**Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click **Add**. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

To edit Issuance Criteria:

- Click **Edit** under Actions for the criteria.

To delete Issuance Criteria:

- Click **Delete** under Actions for the criteria and then click **Save**.
Chapter 7
Service Provider SSO Configuration

Configuring OAuth Attribute Mapping

This configuration allows administrators to use assertion attributes for mapping into
OAuth persistent-grant USER_KEY values (see “About OAuth” on page 10).

**Note:** This option is presented only when enabled for a connection (see
“Choosing IdP Connection Options” on page 359).

To continue, click **Configure OAuth Attribute Mapping**.

**Choosing an OAuth Data Store**

This optional configuration is the same for all OAuth attribute-mapping task flows.
For detailed instructions, see “OAuth Attribute Mapping Using a Data Store” on
page 199.

- If you do not need additional attributes from a data store, just click **Next** on the
  Data Store screen.

**To reach this screen for editing:**

1. Click the connection name on the Main Menu.
   Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the IdP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **OAuth Attribute Mapping** under the Browser SSO tab.
5. Click **Configure OAuth Attribute Mapping**.
6. Click **Data Store** on the Summary screen.

**Configuring Contract Fulfillment**

The last step in configuring OAuth attribute mapping is to map attributes for the
USER_KEY (to look up persistent OAuth grants) and USER_NAME (the name
shown to end-users on permissions screens) for persistent access-token grants.
Map attributes from one of the following Sources:

- **Assertion**
  Values are contained in the assertions from this IdP. When you make this selection, the associated Value drop-down list is populated by the token contract.

- **Context**
  Values are returned from the context of the transaction at runtime.

**Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **JDBC/LDAP/Custom**
  Values are returned from your user-data store. When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes identified for this data store (see “Selecting a Database Table and Columns” on page 376 or “Configuring a Database Filter” on page 378).

- **Expression (when enabled)**
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the assertion, using the `${attribute}` syntax.

  You can also enter values from your data store, when applicable, using this syntax:

  ```
  ${ds.attribute}
  ```

  where `attribute` is any of the data store attributes you have selected.
Chapter 7
Service Provider SSO Configuration

To reach this screen:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click OAuth Attribute Mapping under the Browser SSO tab.
5. Click Configure OAuth Attribute Mapping.
6. Click Contract Fulfillment on the Summary screen.

To map attributes:
1. Choose a Source for each attribute.
   (For descriptions, see “Map attributes from one of the following Sources:” above.)
2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.
3. Click Next.

Selecting Issuance Criteria for OAuth Attribute Mapping

Use this optional screen to define criteria PingFederate can evaluate to determine whether to issue an access token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access a resource.

To configure Issuance Criteria:
1. Select a source containing attributes for token authorization.
   Associated attributes appear in the Attribute Name drop-down list:
   • Assertion – Select to access attributes from the assertion.
   • Context – Select to use values returned from the context of the transaction at runtime.

Note: The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.
Managing IdP Connections

- LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.

**Note:** PingFederate appends a description in parentheses for data stores of the same type.

- Mapped Attributes – Select to access the required attributes from the adapter.

2. Select an attribute name.
3. Select the **Condition** you want to apply.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

4. Enter a value for the attribute.
5. (Optional) Use the **Error Result** field to customize an error code or error message for use if the token authorization fails.

Error results are handled in one of two ways:

- **Redirect** – When an InErrorResource URL is provided, the value of the Error Result field is used by an ErrorDetail query parameter in the redirect URL.
- **Template** – When an InErrorResource URL is not provided, the value of the Error Result field is used by the variable $errorDetail in the sp.sso.error.page.template.html template (for more information on this and other user-facing templates, see “Customizing User-Facing Screens” on page 87).

**Note:** Using an error code in the Error Result field allows the error template or a Web application to process the error result in a variety of ways—for example, a redirect to another page or e-mail to an administrator.

If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click **Add**.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

**Note:** All criteria must pass in order for a user to be authorized.

8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).
Chapter 7

Service Provider SSO Configuration

Important: When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

- Use the in-line editor box to enter the OGNL expression. For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click Add. Continue to add expressions, as needed.
- Click the Test link to input values and test the resulting output for the expression.

Note: For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click Next (or Done, if you are finished with the configuration, and then Save on the Manage Mappings screen).

To edit Issuance Criteria:

► Click Edit under Actions for the criteria.

To delete Issuance Criteria:

► Click Delete under Actions for the criteria and then click Save.

Using the OAuth Attribute Mapping Summary Screen

When you finish the configuration, you can review it on the Summary screen. If you need to make any changes, click the heading over the information you want to edit. When you finish, click Done.

Configuring SAML Protocol Settings

The Protocol Settings screen provides the launching point for configuring bindings, partner endpoints, and other settings needed for the selected SAML profiles (if you are using SAML—see “Choosing SAML Profiles” on page 364). The screen also displays configured information.

(For WS-Federation, the configuration of bindings is not applicable.)
To configure Protocol Settings, you need to know:

- For SP-initiated SSO profiles, the URL(s) of your IdP's Single Sign-on Service(s).
- For SLO profiles, the URL(s) of your IdP's Single Logout Service(s).
- When artifact is an allowable inbound binding, the URL of your IdP's Artifact Resolution Service(s).
- The transport configurations (bindings) that you will use to send and receive data for SSO/SLO connections.
- Digital signature policies and certification requirements to which you and your connection partner have agreed.
- XML encryption policies to which you and your connection partner have agreed.

To continue, click **Configure Protocol Settings**.

**Important:** After modifying any settings, you must click **Save** on the Protocol Settings screen.

**Specifying SSO Service URLs (SAML)**

At this step for SAML 2.0 connections, you associate bindings to the endpoints where your IdP wants PingFederate to send authentication requests when SSO is initiated at your site.

For SAML 1.x, only one endpoint is allowed, and the binding selection is not required.

This configuration applies only to the SP-initiated SSO Profile (see “About SP-Initiated SSO” on page 366).
Service Provider SSO Configuration

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding (SAML 2.0)</td>
<td>The transport type agreed upon by you and your partner: Artifact, POST, or Redirect.</td>
</tr>
<tr>
<td>Endpoint URL</td>
<td>The location where your IdP receives SSO messages.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:

1. Click a connection name on the Main Menu. Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click SSO Service URLs on the Summary screen.
   If this step is not displayed, you have not selected SP-initiated SSO (see “Choosing SAML Profiles” on page 364).

To define an Endpoint URL:

1. If you are using SAML 2.0, select the Binding your partner specifies for the Endpoint.
2. Enter the fully qualified Endpoint URL or a relative path if you have defined a base URL (see “General Connection Information” on page 360).
3. If you are using SAML 2.0, click Add.
4. If your partner has additional SSO endpoints established under SAML 2.0, repeat the steps above.

Specifying a Service URL (WS-Federation)

The Service URL is the WS-Federation endpoint of your IdP partner. This endpoint is where you send RST (Request for Security Token) and SLO messages.
Managing IdP Connections

Enter the fully qualified URL or a relative path if you have defined a base URL (see “General Connection Information” on page 360).

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   - Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
4. Click Protocol Settings under the Browser SSO tab.
5. Click Configure Protocol Settings.
6. Click Service URL on the Summary screen.

Specifying SLO Service URLs

At this step you associate bindings to the endpoints where your IdP receives logout requests when SLO is initiated at your site and where you send SLO responses when you receive SLO requests from the IdP.

This step applies only for SAML 2.0 connections when you have selected an SLO profile (see “Choosing SAML Profiles” on page 364).

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binding</td>
<td>The transport type agreed upon by you and your partner: Artifact, POST, Redirect, or SOAP</td>
</tr>
<tr>
<td>Endpoint URL</td>
<td>The location where your IdP receives SLO request messages.</td>
</tr>
<tr>
<td>Response URL (Optional)</td>
<td>(Optional) The location where the IdP receives logout responses. Use this endpoint when you are part of a chain of session participants.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   - Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click **SLO Service** on the Summary screen.

**To define an Endpoint URL:**
1. Select the Binding your partner specifies for the Endpoint.
2. Enter the fully qualified Endpoint URL or a relative path if you have defined a base URL (see “General Connection Information” on page 360).
3. (Optional) Enter the Response URL or a relative path and click **Add**.
4. If your partner provides additional endpoints for SLO, repeat the steps above.

**Choosing Allowable SAML Bindings (SAML)**

At this step you configure binding(s) that the IdP will use to send SAML assertions or SLO messages (under SAML 2.0) to your PingFederate server.

This configuration applies to all profile types (see “Choosing SAML Profiles” on page 364). You and your partner can agree to standardize on one binding type or select different bindings for different profile scenarios.

---

To reach this screen for editing:
1. Click a connection name on the Main Menu.
2. Click **Manage All IdP**, if needed, to see a full list of connections.
3. Click **Browser SSO** under the IdP Connection tab.
4. Click **Configure Browser SSO**.
5. Click **Protocol Settings** on the Summary screen.
6. Click **Allowable SAML Bindings** on the Summary screen.

**To define binding requirements for this connection:**
- Make your selections and click **Next** (or **Done**).

**Setting an Artifact Lifetime (SAML 2.0)**

When you send an artifact to your IdP's SSO or SLO service, an element in the message indicates how long it should be considered valid.

---

You can change the default value per your requirements, if needed. Also consider synchronizing clocks between your server and your partner's SAML gateway server. If
To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click Artifact Lifetime on the Summary screen.

   This step appears only if you have selected the artifact binding for either a SSO or SLO Service at the IdP site.

Specifying Artifact Resolver Locations

This endpoint or group of endpoints is where your server will send back-channel requests based on artifacts. The location or locations are also known under SAML specifications as the Artifact Resolution Service. SAML 2.0 provides for multiple, indexed endpoints for the service.

Note that this screen is different for SAML 1.x implementations, for which only one endpoint is allowed.

To reach this screen for editing:
1. Click a connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click Browser SSO under the IdP Connection tab.
3. Click Configure Browser SSO.
5. Click Configure Protocol Settings.
6. Click Artifact Resolver Locations on the Summary screen.

   If this step does not appear, you do not have Artifact selected under Allowable SAML Bindings.

For a SAML 2.0 connection:
1. Enter a URL on the Artifact Resolver Locations screen and click Add.
   The URL must be fully qualified (defining protocol, host, and port) unless you have entered a base URL (see “General Connection Information” on page 360).
   Repeat this step if your IdP supports multiple services. The SAML 2.0 specifications permit multiple artifact resolution services through the use of Index numbers, which PingFederate automatically supplies when you add a
Chapter 7
Service Provider SSO Configuration

service. Alternatively, if needed per partner specifications, you may assign these index numbers manually.

**Note:** When specifying multiple artifact resolution endpoints, each endpoint must share the same transport protocol. That is, if one endpoint uses HTTP, then all must use HTTP. Similarly, if one endpoint uses HTTPS, then all must use HTTPS.

2. Click **Next**.

**For a SAML 1.x connection:**

1. Enter the **Endpoint** on the Artifact Resolution Location screen.
   
   The URL must be fully qualified (defining transport protocol, host, and port) unless you have entered a base URL (see “General Connection Information” on page 360).

2. (Optional) Enter your partner’s **Source ID**.
   
   The Source ID is usually a generated value based on a federation partner’s Connection ID; the SP will correctly generate the Source ID. If that is the case for this partner, then leave this field blank. If your partner uses a Source ID that is not based on the Issuer ID, then enter the Source ID supplied by your IdP partner.

3. Click **Next**.

**Configuring Default Target URLs (Optional)**

Use this screen to assign a default target URL for this IdP Connection configuration. Entering a URL in the Default Target URL field overrides any SP Default URL SSO setting (see “Configuring Default URLs” on page 346).

**Note:** SAML 1.x specifications for IdP-initiated SSO require a target URL to be specified. If the target application is specified in the URL parameter (see “IdP Endpoints” on page 529), any URL specified in the Default Target URL field on this screen will not be used for those transactions.

![](image)

**Configuring Signature Policy**

The Signature Policy screen provides options controlling how digital signatures are used for SSO Internet messaging. The choices made on this screen depend on your partner agreement (see “Digital Signing Policy Coordination” on page 28).
Digital signing is required for SAML Response messages sent from the IdP via POST (or Redirect for SAML 2.0). Optionally, SSO authentication requests from the SP (SP-initiated SSO) may also be signed to enforce security. (This option appears only for SAML 2.0 connections and only if you have enabled SP-initiated SSO using the POST or redirect bindings.)

The assertions inside SAML Responses may be also be signed. When you make this choice, only the assertion portion of the Response is signed, not the complete Response. (This is the only option that appears for SAML 1.x connections.)

To choose one or more enhancement options, select the second button, make your selection(s), and click **Next**.

Otherwise, select the first option (if not already selected) and click **Next**.

### Configuring XML Encryption Policy (for SAML 2.0)

For SAML 2.0 configurations, in addition to using signed assertions to ensure authenticity, you and your partner may also agree to encrypt all or part of an assertion to improve privacy. This feature is commonly used if the assertion might pass through an intermediary (such as a user's browser) and HTTPS is not used.

If the name identifier (or `SAML_SUBJECT`) of an assertion is encrypted, you and your partner may also want to encrypt the identifier in subsequent single-logout messages (if you are using an SLO profile).

Note that “The entire assertion” selection on the Encryption Policy screen includes the `SAML_SUBJECT` and all attributes.

To reach this screen for editing:

1. Click a connection name on the Main Menu.
   
   Click **Manage All IdP**, if needed, to see a full list of connections.

2. Click **Browser SSO** under the IdP Connection tab.

3. Click **Configure Browser SSO**.
4. Click **Protocol Settings** under the Browser SSO tab.
5. Click **Configure Protocol Settings**.
6. Click **Encryption Policy** on the Summary screen.

**To define XML encryption:**
1. Select *Allow encrypted SAML Assertions and SLO messages*.
2. Choose whether this IdP partner will encrypt the entire assertion, the `SAML_SUBJECT`, one or more other attributes, or some combination.
3. If your partner is encrypting the name-identifier attribute, use the checkboxes near the bottom of the screen to indicate whether you will encrypt this attribute in outbound SLO messages and/or allow its encryption for inbound messages.

4. Click **Next** or **Done**.

**To disable previously configured XML encryption selections:**
1. Select *None* and then **Done**.
2. Click **Save** on the Browser SSO screen.

**Saving Protocol Settings**

On the Summary screen you can review or edit your Protocol Settings.

---

**Important:** When you finish editing existing settings, be sure to click **Done** on the Summary screen and then **Save** on the Protocol Settings screen. For a new connection, click **Done** and then click **Next** on the Protocol Settings screen. Save the entire connection on the Activation & Summary screen (see “IdP Connection Activation and Summary” on page 433).

---

**To reconfigure saved settings:**
1. Click the heading over the information you want to change.
2. Click **Done** on the screen containing your change.
   
   If you need to make dependent or other changes, do so and continue by clicking **Done** until you reach the Protocol Settings screen.
3. Click **Save** on the Protocol Settings screen.

**Editing and Saving SSO Settings**

On the Summary screen for Browser SSO, you can review or edit your SSO configuration.

---

**Important:** When you finish editing existing settings, be sure to click **Done** on the Summary screen and then **Save** on the Browser SSO screen. For a new connection, click **Done** and then click **Next** on the Browser SSO screen. Save the entire connection on the Activation & Summary screen (see “IdP Connection Activation and Summary” on page 433).

---

**To reconfigure saved settings:**
1. Click the heading over the information you want to change.
2. Click **Done** on the screen containing your change.

   If you need to make dependent or other changes, do so and continue by clicking **Done** until you reach the Browser SSO screen.

3. Click **Save** on the Browser SSO screen.

### Configuring the Attribute Query Option

At the Attribute Query step you configure your connection to request user attributes from your partner IdP, if you have chosen this option (see “Choosing IdP Connection Options” on page 359). Attribute queries are not dependent on single sign-on but may be used independently or in conjunction with Browser SSO or provisioning to provide flexibility in how a user authenticates with SP applications (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

### Setting the Attribute Authority Service URL

**Attribute Authority** is the term used to refer to an IdP that provides user attributes to an **Attribute Requester** (your SP site). The Attribute Authority Service URL corresponds to the endpoint location where Attribute Query requests are received by your IdP partner (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

To configure the URL:

- Enter the fully qualified URL or a relative path if you have defined a base URL (see “General Connection Information” on page 360).

### Mapping Attribute Names

If the application at your site uses different names for user attributes than the names defined by the Attribute Authority, then you need to map them on this screen. When the SP receives a request from a local application to send an Attribute Query to this Attribute Authority partner, the requested user attributes are replaced with the names mapped here.

This information must be predetermined in your agreement with this connection partner.
To map attributes:
1. Enter the Local Name and Remote Name of an attribute and click Add.
   Repeat this step for all attributes requiring mapping.
2. Click Next.

To edit a mapping:
1. Click Edit under Action for the mapping.
2. Make your change(s) and click Update.

Note: If you change your mind, ensure that you click the Cancel link in the Actions column, not the Cancel button, which discards any other changes you might have made in this configuration.

3. Click Done and then Save on the Attribute Query screen.

Defining Security Policy

This screen allows you to specify the digital signing and encryption policy to which you and your partner have agreed. These selections will trigger requirements for setting up Credentials (see “Configuring Security Credentials” on page 424).

This screen also allows you to mask incoming attribute values in log files (see “Attribute Masking” on page 23). When you enable this selection, all user attributes returned from this IdP are masked.

To configure attribute-query security policy for this partner:
► Check or clear the check boxes and click Next or Done.

Saving the Attribute Query Configuration

On the Summary screen you can review the Attribute Query configuration.

To reconfigure saved profiles:
1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.
   If you need to make additional changes, do so and continue by clicking Done until you reach the Attribute Query screen.
3. Click Save on the Attribute Query screen.
Managing IdP Connections

Using Just-in-Time Provisioning

PingFederate's Just-in-Time (JIT) Provisioning allows SPs to create user accounts “on the fly” during SSO events, based on attributes received from IdPs (see “Just-in-Time Provisioning” on page 35). An SP can also use the feature to update existing user records.

To continue, click Configure User Provisioning.

Selecting Attribute Sources (SAML 2.0)

For SAML 2.0 connections, the server can be configured to use only assertion attributes for user provisioning or to retrieve more attributes from the IdP in a follow-on Attribute Query transaction (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started). The User Attributes screen displays the attributes expected in the assertion from this IdP (see “Defining an Attribute Contract” on page 369).

Note: Attribute Query is a SAML 2.0 profile. For SAML 1.x and WS-Federation connections, this screen is not presented: PingFederate uses only attributes from the assertion for user provisioning.

Note: This configuration task is presented in the administrative console only when JIT Provisioning is selected as an option (see “Choosing IdP Connection Options” on page 359).
Chapter 7  
Service Provider SSO Configuration

If you and your IdP partner have agreed to use the Attribute Query profile for provisioning, select that option before leaving this screen.

You configure the Attribute Query profile later in the task flow, if you have not already done so (see “Configuring the Attribute Query Option” on page 401).

Identifying the User Repository

PingFederate’s JIT Provisioning currently supports LDAP v3-compliant user stores and the JDBC-compliant Microsoft SQL Server 2005.

Note: We recommend using the latest version of the Microsoft SQL Server JDBC Driver (version 4 or higher), which is compatible with current and recent JDK versions.

Choose the Active Data Store on the User Repository screen.

If the correct data store is not shown in the drop-down list, then PingFederate is not yet configured to access the store (see “Managing Data Stores” on page 116).

If you are using an LDAP store, refer to the sections immediately following:

- “Specifying an LDAP User-Record Location”
- “Defining an LDAP Filter” on page 405
- “Identifying Provisioning Attributes for LDAP” on page 406

If you are using MSSQL, skip to this section:

- “Selecting a SQL Method” on page 407
### Specifying an LDAP User-Record Location

After choosing a data store, indicate where in the store PingFederate should write new user records or update existing ones. Start by specifying where user records are located in your data store.

**Note:** This screen appears only for LDAP data stores (see “Identifying the User Repository” on page 404).

#### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. Leave this field blank if records are located at the LDAP root.</td>
</tr>
</tbody>
</table>

### Defining an LDAP Filter

On the Unique ID screen, create an LDAP filter to identify user accounts to be provisioned (or updated) during SSO events. PingFederate uses this expression in conjunction with the Base DN (see the previous section) to locate existing account records and to add new ones.

**Note:** This screen appears only for LDAP data stores (see “Identifying the User Repository” on page 404).
The filter is in the form:
attribute=${value}

Note that the statement must not be enclosed in parentheses, unlike filters used to retrieve LDAP attributes for adapter mapping (see “Specifying an LDAP Filter” on page 381).

The left-side variable is an attribute in your user-data store—click the link near the left corner of the screen to see a list of available attributes. The right side of the filter generally uses one or more attribute values passed in from the assertions (see “Defining an Attribute Contract” on page 369). Variables for these attributes, including the correct syntax, are listed under Assertion Values.

**Note:** If you are unfamiliar with writing LDAP queries, please refer to the documentation accompanying your LDAP installation.

### Identifying Provisioning Attributes for LDAP

On the Attributes screen, select the data-store attributes to be provisioned.

**Tip:** Multiple-value IdP attributes are handled automatically: when you map a multi-value assertion attribute to an LDAP attribute, each value is stored separately for the LDAP attribute name. If you need to provide additional values for particular attributes, add the same attribute name to this list multiple times. You can then map the additional values on the Attribute Fulfillment screen (see “Mapping Attributes to a User Account” on page 410).
Managing IdP Connections

To select attributes:

1. Choose a Root Object Class and an Attribute from the drop-down lists and click Add Attribute.

   Repeat this step for the same attribute if needed for multi-value attributes (see “Tip” above).

   **Important:** For the Oracle Directory Server or Active Directory, the attribute objectClass must be among attributes added—select <Show All Attributes> under Root Object Class to locate and add this attribute.

2. Repeat the previous step for each attribute requiring provisioning.

Selecting a SQL Method

For JDBC data stores, PingFederate allows you to map attributes directly to a single database table (the default) or to SQL stored-procedure parameters.

**Note:** The SQL Method screen appears only for JDBC data stores (see “Identifying the User Repository” on page 404).

Make a selection as needed and click Next.

Depending on the selection, different steps appear under the JIT Provisioning task. Refer to the manual sections indicated below for more information:
Chapter 7
Service Provider SSO Configuration

If you are mapping attributes directly to a Table, refer to the sections immediately following:

* “Specifying a Database User-Record Location” (next)
* “Specifying a Unique-ID Database Column” on page 409

If you are using a Stored Procedure, skip to this section:

* “Specifying a Stored-Procedure Location” on page 409

### Specifying a Database User-Record Location

For database provisioning to a table, indicate where PingFederate should write new user records or update existing ones.

**Note:** This screen appears only for MSSQL data stores when Table is selected on the SQL Method screen (see the previous section, “Selecting a SQL Method”).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Select the table structures that store information within the database.</td>
</tr>
<tr>
<td>Table</td>
<td>Select the name of the database table that contains user records.</td>
</tr>
<tr>
<td>Columns to fulfill when provisioning</td>
<td>For the selected table, all attributes and their data types are displayed. All attributes must be mapped for the database insertion to succeed, although a null entry may be used for optional attributes (see “Mapping Attributes to a User Account” on page 410).</td>
</tr>
</tbody>
</table>
Managing IdP Connections

Specifying a Unique-ID Database Column

PingFederate uses the column specified on this screen to check whether a user record already exists for the incoming assertion.

**Note:** This screen appears only when Table is selected on the SQL Method screen (see “Selecting a SQL Method” on page 407).

- Select a column that represents a unique characteristic about the database entry for a particular user (email, for example).

### Specifying a Stored-Procedure Location

If you are using a stored procedure for provisioning the user database, specify its location on this screen.

**Important:** The database account used by PingFederate must have access to the schema in which the stored procedure is located, as well as execute permission for the procedure.

**Note:** This screen appears only when Stored Procedure is selected on the SQL Method screen (see “Selecting a SQL Method” on page 407).
## Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Select the table structure that contains stored procedures within the database.</td>
</tr>
<tr>
<td>Stored Procedure</td>
<td>Select the stored procedure needed to provision the user database.</td>
</tr>
<tr>
<td>Procedure parameters to fulfill</td>
<td>For the selected procedure, all parameters and their data types are displayed. You map assertion values to the parameters on the next screen.</td>
</tr>
</tbody>
</table>

▶ To see a list of attributes expected from SAML assertions, click View Attribute Contract.

▶ If the list of parameters is not or might not be current (due to any recent procedure revisions), click Refresh.

### Mapping Attributes to a User Account

The Attribute Fulfillment screen provides a means of mapping the values of incoming attributes into local account attributes or, for JDBC stores, into SQL stored-procedure parameter values (see “Selecting a SQL Method” on page 407). You can also provide values of your own for any data store attributes (except JDBC system-managed) or for SQL procedure parameters—either as hard-coded text or derived values based on assertion attributes.

For JDBC, this screen also provides a means of testing the insertion of attribute values into the database or stored procedure.

**Tip:** For mapping to a database `datetime` or `smalldatetime` column, if you are not using a stored procedure to convert the incoming `string` value, you may use a PingFederate Java conversion method via OGNL expressions (see “Using Attribute Mapping Expressions” on page 573). Note that expressions must be enabled to use the PingFederate conversion method—see “Enabling and Disabling Expressions” on page 573.

**Note:** For a JDBC data store, if stored procedures are used, the note under the task bar on this screen is different than shown above and the Target Attribute column reads “Target Parameter.”
Map attributes from one of these Sources:

- **Assertion**
  Values are contained in the assertions from this IdP. When you make this selection, the associated Value drop-down list is populated by the attribute contract (see “Defining an Attribute Contract” on page 369).

- **Context**
  Values are returned from the context of the transaction at runtime—for example, authentication context.

- **Attribute Query**
  This choice appears only if you have chosen to use the Attribute Query profile for provisioning (see “Selecting Attribute Sources (SAML 2.0)” on page 403).
  To map an attribute-query value, use this syntax:
  ```$\{query\_attribute\}```
  You can also combine attribute-query values with references to attributes in the attribute contract. For example:
  ```$\{query\_attribute\}+$\{attribute\}```
  References to attributes not contained in the attribute contract result in an Attribute Query back to the IdP partner.

- **Expression** (when enabled—see “Enabling and Disabling Expressions” on page 573)
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

  **Tip:** For JDBC mapping, if the data type of a Target Attribute/Parameter is `datetime` or `smalldatetime`, you can use an expression to convert date-time strings from the assertion. After selecting Expression for such attributes, click **Datetime OGNL**

  **Examples** under the text box for syntax information and examples.

- **System Managed (if applicable)**
  This mapping option appears only when any automatically assigned JDBC attributes are among columns to be provisioned—for example, an identity or timestamp column for the MSSQL Server.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the assertion, using the `$\{attribute\}$` syntax.

  **Note:** If no entry is required in a JDBC database for a column, you can leave the text box blank. A blank entry results in an empty string in the database for string data types and `null` for all other data types. Alternatively, for string types, you can enter `null` in the text box to explicitly set `null` in the table column.
To map attributes:

1. Choose a Source for provisioning each Target Attribute or Target Parameter (see descriptions of each Source type above).

   **Note:** For JDBC table mapping, select System Managed as the Source for any columns that are automatically provisioned by the database server.

   **Tip:** For LDAP mapping, choose Text as the Source for the objectClass attribute.

2. Choose (or enter) a Value for each Attribute.

   All values must be mapped. However, for optional JDBC table columns, you may leave a text box blank (or, for string data types, enter `null` to avoid empty strings).

   Note that no value is required for System Managed attributes.

   **Tip:** For Active Directory, enter `user` in the text box for objectClass. For the Oracle Directory Server, enter `inetOrgPerson`.

3. Click **Done**.

To test the insertion of attributes into a JDBC table or stored procedure:

1. Click the **Test insert into . . .** or **Test call to . . .** link.

2. For each Target Attribute or Target Parameter (for a stored procedure), enter text into the boxes under **Test Value** and click **Test Insert** (or **Test Stored Procedure Call**).

   For table insertions, if the test is successful, a confirmation is displayed along with the values inserted. For stored procedures, only a confirmation is displayed if the test is successful, indicating that the procedure was populated with parameter values.

3. For table insertions, unless you wish to keep test values in the database, click **Roll Back All Test Inserts**.

   If you are using a stored procedure, this option is not provided since PingFederate cannot know the result of the procedure. Database rollback, if needed, must be handled manually.

4. When finished, click **Next** or **Done**; or click the link **Return to Attribute Fulfillment** to continue or change any mapping.

Choosing an Event Trigger

On the Event Trigger screen, choose whether PingFederate initiates user provisioning only when the user identifier is new or every time your site receives a SAML assertion. In the latter case (for all assertions), an existing user account is always updated with incoming attributes.

**Note:** This screen does not appear for JDBC data stores if provisioning is accomplished using a stored procedure (see “Selecting a SQL Method” on page 407), because the procedure is always called for all assertions. The procedure should handle both provisioning new users and updating existing ones (if desired).
Managing IdP Connections

Error Handling

If user provisioning fails for any reason during SSO events, you can choose to stop the SSO or continue the process by passing assertion attributes on to your application (via the SP adapter configuration—see “Configuring Adapter Mapping and User Lookup” on page 370).

When SSO is aborted, the user is redirected to an error page, and the failure is written to the log and added to the Transaction Failure Count at the bottom of the administrative console.

Using the Provisioning Summary Screen

The Summary screen provides an overview of your provisioning configuration.

To change the configuration:
1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.
   If you need to make additional changes, do so and continue by clicking Done until you reach the JIT Provisioning screen.
3. Click Save on the JIT Provisioning screen.

Configuring Inbound Provisioning

Inbound Provisioning implements the SCIM protocol to provide an automated user-management service when PingFederate is configured as an SP (see “Provisioning for SPs” on page 35).

This configuration provides for two-way mapping of attributes. The first facilitates SCIM operations used to create and update records in the data store (see “Writing User Information to the Data Store” on page 417). The second allows the same SCIM client to retrieve those records and have the attribute values mapped back to their corresponding designation in the client store (see “Configuring a SCIM Response” on page 420).
The dual mapping is intended to provide greater flexibility, especially when needed for OGNL-expression transformations—for example, converting two attributes into one multi-value attribute and then back again (see “Using Attribute Mapping Expressions” on page 573).

**Note:** SCIM-client requests must include authentication credentials, which you configure later (see “Configuring Security Credentials” on page 424). The same credentials needed for SSO or other types of transactions enabled as part of this IdP connection, if configured, are also used for SCIM transactions.

To continue, click **Configure Inbound Provisioning**.

### Specifying the User Repository

PingFederate currently supports AD user stores (requires an SSL connection to the user store) as well as custom identity store provisioners for Inbound Provisioning.

- Choose either **Active Directory Data Store** or **Identity Store Provisioner** and then specify the data store from the drop-down menu.

**Note:** If the correct data store is not shown in the drop-down list, then PingFederate is not yet configured to access the store (see “Managing Data Stores” on page 116).

If the identity store provisioner is not shown in the drop-down list, then PingFederate is not yet configured to access the store (see “Configuring Identity Store Provisioners” on page 344).
Identifying an LDAP User-Record Location

After choosing a data store, indicate where in the data store user records exist so PingFederate can create, read, or update them.

Enter the base Distinguished Name of the tree structure where user records are stored. PingFederate looks only at this node level or below it for user accounts that need provisioning.

Defining a Unique ID

On the Unique ID screen, create an LDAP filter to resolve user accounts for SCIM operations. PingFederate uses this expression in conjunction with the Base DN (see the previous section) to add new account records.
Chapter 7
Service Provider SSO Configuration

Filter

CN=$(name,givenName)

SCIM Attributes

$[active]
$[addresses.home.country]
$[addresses.home.formatted]
$[addresses.home.isPrimary]
$[addresses.home.locality]
$[addresses.home.postalCode]
$[addresses.home.region]
$[addresses.home.streetAddress]
$[addresses.home.type]
$[addresses.other.country]
$[addresses.other.formatted]
$[addresses.other.isPrimary]
$[addresses.other.locality]
$[addresses.other.postalCode]
$[addresses.other.region]
$[addresses.other.streetAddress]
$[addresses.other.type]
$[addresses.work.country]
$[addresses.work.formatted]
$[addresses.work.isPrimary]
$[addresses.work.locality]
$[addresses.work.postalCode]
$[addresses.work.region]
$[addresses.work.streetAddress]
$[addresses.work.type]
$[displayName]
$[emails.home.isPrimary]
$[emails.home.type]
$[emails.home.value]
$[emails.other.isPrimary]

Note: This screen appears only if you are configuring an LDAP user store for provisioning.

The filter is in the form:
attribute=${value}

Note: Unlike filters used to retrieve LDAP attributes for adapter mapping, do not enclose the statement in parentheses (see “Specifying an LDAP Filter” on page 381).

The left-side variable is an attribute in your user-data store—click the link near the left corner of the screen to see a list of available attributes. The right side of the filter generally uses one or more attribute values passed in from the SCIM request. Variables for these attributes, including the correct syntax, are listed under SCIM Attributes.

Note: If you are unfamiliar with writing LDAP queries, please refer to the documentation accompanying your LDAP installation.

To construct the LDAP filter:
1. Enter the statement in the space provided, following the guidelines and example above.
2. Ensure the syntax and variable names are correct.
3. Click Next.

Writing User Information to the Data Store

To configure how PingFederate completes create and update operations for user accounts from a SCIM request, identify incoming attributes and map them to data-store attributes.

Note: Unlike filters used to retrieve LDAP attributes for adapter mapping, do not enclose the statement in parentheses (see “Specifying an LDAP Filter” on page 381).

If you are unfamiliar with writing LDAP queries, please refer to the documentation accompanying your LDAP installation.

Click Configure Write Users to continue.

Identifying Inbound Provisioning Attributes for LDAP

On the Attributes screen, select the data-store attributes you want to provision.
System Managed LDAP Attributes

Several attributes are managed internally by PingFederate and do not require mapping:
- objectClass
- unicodePwd
- objectGUID
- userAccountControl

You can override the internal management of objectClass and unicodePwd by selecting these attributes and mapping them to SCIM attributes on the Attribute Fulfillment screen. In this case, the values you supply are used. The objectGUID and userAccountControl attributes cannot be overridden and are ignored if selected.

**To select attributes:**
1. Choose a Root Object Class and an Attribute from the drop-down lists and click Add Attribute.
2. Repeat the previous step for each attribute requiring provisioning.

**Mapping Attributes to User Accounts**

Use this screen to map attribute values in the SCIM request to user-account attributes.
Managing IdP Connections

Map attributes from one of these sources:

- **Context**
  Values are returned from the context of the transaction at runtime.

  **Note:** The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **Expression** (when enabled—see “Enabling and Disabling Expressions” on page 573)
  This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

  **Tip:** If two attribute values from a SCIM request need to be mapped to one LDAP attribute value, use an OGNL Expression to create it.

- **SCIM Request**
  When you make this selection, the associated Value drop-down list is populated by defined components of the SCIM request.

- **Text**
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the SCIM request, using the \$\{attribute\} syntax.
To map attributes:
1. Choose a Source for provisioning each Target Attribute (see descriptions of each Source type above).
2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.
3. Click Done.

Using the Write Users Summary Screen

The Summary screen provides an overview of the inbound provisioning configuration for request mapping.

► When you finish with a new configuration, click Done.

To change the configuration:
1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.

   If you need to make additional changes, do so and continue by clicking Done until you reach the Writer Users screen.
3. Click Next on the Inbound Provisioning screen.

Configuring a SCIM Response

To configure a SCIM response to a request to read and return provisioned SCIM attributes, identify and map the user account attributes you want to include.

► Click Configure Read Users to continue.

Identifying Expected User Attributes for the SCIM Response

An attribute contract is a set of user attributes that you and your partner have agreed will be sent in a SCIM response for this connection. The attributes you mapped to user account attributes in the Write Users flow appear at the top of the screen.

Use the Available SCIM Attributes link at the bottom of the screen to include additional attributes you want to map in the SCIM response.

Optionally, you can mask the values of attributes in the log files that PingFederate writes when it sends the SCIM response.

System Managed SCIM Attributes

There are several SCIM attributes that are managed internally by PingFederate and are unavailable for inclusion in the attribute contract:

- **id**
Managing IdP Connections

- active

To add an attribute:
1. Enter the attribute name in the text box.
   
   Attribute names are case-sensitive and must correspond to the attribute names expected by your partner. To see a list of available attributes, click Available SCIM attributes.
2. (Optional) Select the checkbox under Mask Values in Log.
3. Click Add.

To modify an attribute name or masking selection:
1. Click Edit under Action for the attribute.
2. Make the change and click Update.

Note: If you change your mind, ensure that you click Cancel in the Actions column, not the Cancel button, which discards any other changes you might have made in the configuration steps.

To delete an attribute:
1. Click Delete under Action for the attribute.

Identifying LDAP Attributes for the SCIM Response

Select the LDAP attributes you want to map to attributes in the SCIM response.

<table>
<thead>
<tr>
<th>Attribute Contract</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT OBJECT CLASS</td>
<td>ACTION</td>
</tr>
<tr>
<td>mail</td>
<td>Remove</td>
</tr>
<tr>
<td>title</td>
<td>Remove</td>
</tr>
<tr>
<td>sn</td>
<td>Remove</td>
</tr>
<tr>
<td>initials</td>
<td>Remove</td>
</tr>
<tr>
<td>telephoneNumber</td>
<td>Remove</td>
</tr>
<tr>
<td>givenName</td>
<td>Remove</td>
</tr>
<tr>
<td>displayName</td>
<td>Remove</td>
</tr>
<tr>
<td>mobile</td>
<td>Remove</td>
</tr>
</tbody>
</table>

Note: This screen appears only if you are configuring an LDAP user store for provisioning.

To select attributes:
1. Choose a Root Object Class and an Attribute from the drop-down lists and click Add Attribute.
2. Repeat the previous step for each attribute requiring provisioning.
Mapping Attributes to SCIM Response Attributes

Use this screen to map outgoing user-account attributes to SCIM responses to READ requests.

Map attributes from one of these sources:

- **Context**
  
  Values are returned from the context of the transaction at runtime.

  Note: The HTTP Request selection is retrieved as a Java object rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.
  
  Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

- **Expression** (when enabled—see “Enabling and Disabling Expressions” on page 573)

  This option provides more complex mapping capabilities—for example, transforming outgoing values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

  Tip: If an LDAP attribute needs to be mapped to two attributes in a SCIM response, use an OGNL Expression to create them.

- **LDAP**

  Values are returned from your query. When you make this selection, the Value list is populated by the LDAP attributes you identified for this data store.

- **Identity Store**

  Values are returned from your query. When you make this selection, the Value list is populated by the Identity Store attributes you identified for this data store.

<table>
<thead>
<tr>
<th>TARGET ATTRIBUTE</th>
<th>SOURCE</th>
<th>VALUE</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayname</td>
<td>LDAP</td>
<td>displayName</td>
<td>None</td>
</tr>
<tr>
<td>emails.work-value</td>
<td>LDAP</td>
<td>mail</td>
<td>None</td>
</tr>
<tr>
<td>name.familyname</td>
<td>LDAP</td>
<td>sn</td>
<td>None</td>
</tr>
<tr>
<td>name.givenName</td>
<td>LDAP</td>
<td>givenName</td>
<td>None</td>
</tr>
<tr>
<td>name.middleName</td>
<td>LDAP</td>
<td>initials</td>
<td>None</td>
</tr>
<tr>
<td>phoneNumbers.mobile.value</td>
<td>LDAP</td>
<td>mobile</td>
<td>None</td>
</tr>
<tr>
<td>phoneNumbers.work-value</td>
<td>LDAP</td>
<td>telephoneNumber</td>
<td>None</td>
</tr>
<tr>
<td>title</td>
<td>LDAP</td>
<td>title</td>
<td>None</td>
</tr>
</tbody>
</table>
Managing IdP Connections

- Text
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the SCIM request, using the $\{attribute\}$ syntax.

To map attributes:
1. Choose a Source for provisioning each Target Attribute (see descriptions of each Source type above).
2. Choose (or enter) a Value for each Attribute.
   All values must be mapped.
3. Click Done.

Using the Read Users Summary Screen
The Summary screen provides an overview of the inbound provisioning configuration for SCIM response mapping.

- When you finish with a new configuration, click Done.

To change the configuration:
1. Click the heading over the information you want to change.
2. Click Done on the screen containing your change.
   If you need to make additional changes, do so and continue by clicking Done until you reach the Read Users screen.
3. Click Save on the Inbound Provisioning screen.

Handling SCIM Delete Requests
Use this screen to define how SCIM delete requests are handled within your user data store.

Select Disable User to make the user inactive within the data store. This approach might be preferred in situations where accounts must be retained for auditing reasons.

In order to be SCIM compliant when deleting users, PingFederate returns an HTTP 404 response code for all subsequent operations related to the user—effectively treating the user as if they have been deleted from the LDAP user store (see the SCIM protocol).

Caution: If the user is disabled through another method, PingFederate still treats that user as if they have been deleted and returns HTTP 404 response codes for all subsequent requests.
chapter 7
service provider sso configuration

- Select **Permanently Delete User** to remove the user from the data store.

**Saving the Inbound Provisioning Configuration**

On the Summary screen you can review the Inbound Provisioning configuration.

To **reconfigure saved profiles**:

1. Click the heading over the information you want to change.
2. Click **Done** on the screen containing your change.
   
   If you need to make additional changes, do so and continue by clicking **Done** until you reach the Inbound Provisioning screen.
3. Click **Save** on the Inbound Provisioning screen.

**Configuring Security Credentials**

The **Credentials** screen presents a list of possible security requirements you might need, depending on the federation protocol you are using and the choices you have made.

- To configure or modify credentials, click **Configure Credentials**.

**Back-Channel Authentication**

When you configure a profile for the **inbound** artifact binding, **outbound SOAP** binding, or provisioning, you must specify back-channel authentication information for sending SOAP messages, artifact resolution requests, and provisioning requests to your partner IdP.

Similarly, if you send artifacts, SOAP messages, or provisioning messages to your partner IdP, then you must configure SOAP authentication requirements for receiving SOAP responses, artifact resolution requests, or provisioning requests from your partner.

This step also applies to attribute-request configurations, since this profile always uses the SOAP back channel (see “Choosing SAML Profiles” on page 364).
### Managing IdP Connections

To reach this screen for editing:

1. Click a connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Credentials** under the **IdP Connection** tab.
3. Click **Configure Credentials**.

   If the Back-Channel Authentication step is not shown, then it is not applicable to your configuration—you are not using the Attribute Query profile and have not configured any profiles to use the artifact or SOAP bindings.

To configure back-channel authentication requirements for sending HTTP messages:

1. On the Back-Channel Authentication screen, click the **Configure** link to the right of the list of messages to be sent to your partner.
2. Make one or more selections on the Outbound SOAP Authentication Type screen:
   - **HTTP Basic** — you enter SOAP Basic credentials on a later screen.
   - **SSL Client Certificate** — you specify the certificate on a later screen.
     - This option is available only if you specify an endpoint that uses SSL.
   - **Use Digital Signatures (Browser SSO profile only)** — you sign the message.
     - You are asked to select a signing certificate on a later screen.

   For SAML 2.0, these options may be used in any combination or independently. For SAML 1.x, you must use either Basic or SSL authentication; digital signing may be added to ensure message integrity.

   By default, PingFederate validates your partner’s SSL server certificate—verifying that the certificate chain is rooted by a trusted Certificate Authority and that the hostname matches the certificate’s Common Name. Clear the associated checkbox if you do not want this validation to occur.

3. Click **Next**.
4. If you chose HTTP Basic at Step 2, enter the SOAP Username and Password to use for this partner under Basic SOAP Authentication.
   - You must obtain these credentials from your partner.

---

**Note:** A yellow triangle next to a listing indicates that you have not completely configured back-channel authentication requirements.
5. If you are using an SSL certificate, select the certificate under SSL Authentication Certificate and click Next.
   If you have not yet created or imported the client SSL certificate you need into PingFederate, click Manage Certificates (see “SSL Client Keys and Certificates” on page 207). You need to export the certificate (only) and send it to your partner.

6. On the Summary screen, click Done.

**To configure back-channel authentication requirements for receiving HTTP messages:**

1. On the Back-Channel Authentication screen, click the Configure link to the right of the list of messages to be received from your partner.

2. Select one or more options on the Inbound Authentication Type screen:
   - HTTP Basic — Enter the logon username and password your partner uses on the next screen.
   - SSL Certificate — Specify certificate verification information on a later screen.
   - Use Digital Signatures (Browser SSO profile only). . . — Incoming messages must be signed.
   - Require SSL — When selected, incoming HTTP transmissions must use a secure channel.
     You are asked to select a signature verification certificate on a later screen.
     For SAML 2.0, use these options in any combination or independently. For SAML 1.x, you must use either Basic or SSL authentication; add digital signing to ensure message integrity.

3. Click Next.

4. If you chose HTTP Basic at Step 2, enter the Username and Password under Basic Authentication (Inbound).

   **Important:** If you are configuring more than one connection that uses the artifact or HTTP profile, you must ensure that the Username is unique for each connection.

5. If you are using an SSL certificate, select Anchored or Unanchored under Certificate Verification Method.
   - Anchored — The certificate must be signed by a trusted Certificate Authority, and the CA's certificate must be imported into the PingFederate Trusted CA store (see “Trusted Certificate Authorities” on page 204).
   - Unanchored — The certificate is self-signed or you want to trust a specified certificate.

   **Note:** When anchored certificates are used between partners, certificates may be changed without sending the update to your partner. If the certificate is unanchored, any changes must be promulgated.

6. Click Next.
7. If you chose anchored SSL certificate verification at Step 5, enter the Subject DN and click **Next**.

   **Tip:** If you have not yet defined the certificate in PingFederate or you do not know the DN, return to the previous screen and select Unanchored. Then click **Next** and click **Manage Certificates** on the SSL Verification Certificate screen to import the certificate, if needed, or to view its DN.

8. If you chose unanchored SSL certificate verification at Step 5, select the certificate to use for validating the SSL connection.

   If you have not yet imported the certificate into PingFederate, click **Manage Certificates**.

9. Click **Next**.

10. On the Summary screen, click **Done**.

### Digital Signature Settings

This step defines the private key you will use to sign SSO authentication or attribute requests (optionally) or SAML 2.0 SLO messages for this IdP. In addition, the step allows you to include “Key Info” with the XML message if you and your partner have agreed to this option.

Digital signing applies to SP-initiated SSO under SAML 2.0, when specified by your partner agreement, and to either SLO profile (see “Choosing SAML Profiles” on page 364) using the POST or redirect bindings. The step also applies if you are configuring an Attribute Query profile and have specified that you will sign attribute requests (see “Defining Security Policy” on page 402).

The step is not required for SAML 1.x IdP connections.

To reach this screen for editing:

1. Click a connection name on the Main Menu.

   Click **Manage All IdP**, if needed, to see a full list of connections.

2. Click **Credentials** under the IdP Connection tab.

3. Click **Configure Credentials**.

4. Click **Digital Signature Settings** on the Summary screen.

   If this step does not appear, then your configuration does not require digital signatures. You do not have SLO configured using the POST or redirect bindings, and you have not elected to sign either authentication or attribute
requests (see “Configuring Signature Policy” on page 398 and “Defining Security Policy” on page 402).

To specify a certificate:

1. Select the certificate from the drop-down list.
   
   If you have not yet created or imported your certificate into PingFederate, click Manage Certificates (see “Digital Signing and Decryption Keys and Certificates” on page 209).
   
2. (Optional) If you have agreed to send your public key with the SAML message, select the checkbox to include the certificate. To include the raw key in the signature as well, select the Include the raw key ... checkbox.
   
3. (Optional) Select the Signing Algorithm from the drop-down list.
   
   The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

Signature Verification Settings

Under SAML 2.0 specifications, when your site receives any SAML 2.0 messages via the POST or Redirect bindings, the messages must be digitally signed. Signing is also always required for the SAML 1.x POST binding and for WS-Federation assertions, as well as incoming SAML 1.1 or 2.0 tokens for WS-Trust STS processing.

Depending on your agreement with this IdP, SSO assertions, SAML 2.0 artifacts, or SOAP messages might also require signatures.

Whenever signatures are required, PingFederate provides a choice of trust models, including an option to use anchored signature-verification certificates embedded in incoming messages (see “Trust Models” on page 28). When this option is chosen in Signature Verification Settings, you must provide the Subject DN for embedded certificates coming from this partner, and the Issuer CA certificate must be part of the PingFederate trusted store (see “Trusted Certificate Authorities” on page 204).

Alternatively, you may choose to use unanchored certificates, in which case you must import your partner’s public-key certificate during this configuration (or select it if it is already imported). To prevent any interruption of service due to an expired certificate, you can ask your partner for a new certificate in advance and import it as backup.

To continue, click Manage Signature Verification Settings.

To reach this screen for editing:

1. Click a connection name on the Main Menu.
   
   Click Manage All IdP, if needed, to see a full list of connections.
   
2. Click Credentials under the IdP Connection tab.
   
3. Click Configure Credentials.
4. Click **Signature Verification Settings** on the Summary screen. If this step does not appear, then your configuration does not require verification settings.

**Selecting a Trust Model**

This screen allows you choose the Trust Model you want to use for signature verification (see “Trust Models” on page 28).

- Depending on the selection, the next step in this task varies:
  - For **Anchored**, the next step is to enter the Subject DN for your partner's certificate (see the next section, “Specifying a Subject DN”).

  **Important:** If you are using the Redirect binding for SLO, you cannot use anchored certificates because SAML 2.0 does not permit certificates to be included using this transport method.

  - For **Unanchored**, the next step is to import your partner's certificate (see “Selecting an Unanchored Certificate” on page 430).

**To reach this screen for editing:**

1. Click a connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Credentials** under the IdP Connection tab.
3. Click **Configure Credentials**.
4. Click **Signature Verification Settings** on the Summary screen.
   - If this step does not appear, then your configuration does not require verification settings.
5. Click **Manage Signature Verification Settings**.
6. Click **Trust Model** on the Summary screen.

**Specifying a Subject DN**

When you choose to use an anchored certificate for signature verification, incoming SAML messages must contain the partner's verification certificate (see “Trust Models” on page 28). PingFederate verifies that the Issuer CA is trusted and checks to see that the embedded certificate's Subject DN matches the one specified on this screen. If so, PingFederate uses that certificate to verify the message signature.
Chapter 7
Service Provider SSO Configuration

You can either enter the Subject DN or extract it from your partner's certificate if the certificate is stored on an accessible file system.

**Important:** Ensure that you enter the Subject DN correctly or extract the DN from the correct partner certificate.

**To extract the Subject DN from a certificate:**

1. Click browse to select your IdP partner's public certificate.
2. Click **Extract**.

**To reach this screen for editing:**

1. Click a connection name on the Main Menu.
2. Click **Manage All IdP**, if needed, to see a full list of connections.
3. Click **Credentials** under the IdP Connection tab.
4. Click **Configure Credentials**.
5. Click **Signature Verification Settings** on the Summary screen.
   - If this step does not appear, then your configuration does not require verification settings.
6. Click **Manage Signature Verification Settings**.
7. Click **Certificate Subject DN** on the Summary screen.

**Selecting an Unanchored Certificate**

On the Signature Verification Certificate screen, you identify your partner's imported public certificate and, optionally, a secondary certificate to use when the first expires (see “Trust Models” on page 28).

**To reach this screen for editing:**

1. Click a connection name on the Main Menu.
2. Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Credentials** under the IdP Connection tab.
3. Click **Configure Credentials**.
4. Click **Signature Verification Settings** on the Summary screen.
   - If this step does not appear, then your configuration does not require verification settings.
5. Click **Manage Signature Verification Settings**.
6. Click **Signature Verification Certificate** on the Summary screen.

**To specify a verification certificate:**
1. Select the certificate from the drop-down list.
   - If you have not yet imported the certificate into PingFederate, click **Manage Certificates**.
2. (Optional) Select a Secondary certificate for backup.
   - Use this field if your partner has sent you a new certificate to replace one that is ready to expire. The server will automatically verify against the secondary certificate when the primary one expires.

**Choosing an Encryption Certificate**

If `SAML_SUBJECT` is encrypted, either by itself or as part of a whole assertion, then all references to this name identifier in SAML 2.0 SLO requests from your site may also be encrypted (if the connection uses SP-initiated SLO). For more information, see “Configuring XML Encryption Policy (for SAML 2.0)” on page 399.

To enable this XML encryption, you must identify an encryption certificate for this partner.

You must also choose a certificate if encryption of the Name Identifier is required for an Attribute Request profile (see “Defining Security Policy” on page 402).

**To reach this screen for editing:**
1. Click a connection name on the Main Menu.
2. Click **Manage All IdP**, if needed, to see a full list of connections.
3. Click **Credentials** under the IdP Connection tab.
4. Click **Select XML Encryption Certificate** on the Summary screen.
   - If this step is not present, then you have either not configured this connection to use the SP-initiated SLO profile (see “Choosing SAML Profiles” on page 364) or you have chosen not to encrypt the assertion or the `SAML_SUBJECT` (see “Configuring XML Encryption Policy (for SAML 2.0)” on page 399).
To identify the encryption certificate:

1. (Optional) Change the default settings under Block Encryption Algorithm.

   Due to import control restrictions, the standard JRE distribution supports strong but not unlimited encryption. To use the strongest AES encryption, when permissible, download and install the appropriate version of “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” from the Oracle download Web site (www.oracle.com/technetwork/java/javase/downloads/index.html).

   For more information about XML block encryption and key transport algorithms, see the “XML Encryption Syntax and Processing W3C Recommendation” (http://www.w3.org/TR/xmlenc-core/).

   **Note:** As a Key Transport Algorithm, RSA-v1.5 is disabled for new connections for security reasons. If you are updating an existing connection that uses RSA-v1.5, we recommend changing the selection for increased security.

Choosing a Decryption Key

As part of XML encryption, you must identify a certificate and key for PingFederate to use to decrypt incoming assertions or assertion elements (see “Configuring XML Encryption Policy (for SAML 2.0)” on page 399).

To reach this screen for editing:

1. Click a connection name on the Main Menu.

   Click Manage All IdP, if needed, to see a full list of connections.

2. Click Credentials under the IdP Connection tab.

3. Click Configure Credentials.

4. Click Select XML Decryption Key.

   If this step is not present, you have not chosen to require encryption of all or part of the SAML assertion (see “Configuring XML Encryption Policy (for SAML 2.0)” on page 399).

To identify the decryption key:

- From the drop-down list, select the applicable certificate and click Next.

If the certificate is not in the list, click Manage Certificates to import it (see “Digital Signing and Decryption Keys and Certificates” on page 209).
Saving Credential Configurations

From the Summary screen you can review or edit your credentials configuration.

**Important:** When you finish editing existing settings, you must click **Done** on the Summary screen and then **Save** on the Credentials screen. For a new connection, click **Done** and then click **Next** on the Credentials screen. Save the entire connection on the Activation screen (see “IdP Connection Activation and Summary” next).

IdP Connection Activation and Summary

When you finish setting up a connection, you may choose to activate it immediately.

**Important:** Regardless of whether you choose to activate a new connection now or later, you must click **Save** on the Summary screen for a new connection if you want to keep the configuration.

You can deactivate a connection at any time (for maintenance, for example). When a connection is inactive, all SSO or SLO transactions to or from this partner are disabled, as well as access to the WS-Trust STS for Web Service Providers associated with this connection.

**Tip:** The SSO Application Endpoint near the top of the Summary screen is an example URL that webmasters or Web application developers at your site might use to invoke SSO for the connection. For details about SSO and other server endpoints, including optional query parameters, see “Application Endpoints” on page 529.

To change a Connection Status:

- Select either Active or Inactive and then click **Save**.

To modify a connection setting:

1. If you know which step needs to be modified, click its link under the IdP Connection tab.

   If you do not know where to change the setting, locate the currently configured data under one of the summary headings and then click the subheading above the data.

2. Change the information on the step screen and click **Save**, if available.

   If **Save** is not available, you are in the middle of a task (see “About Tasks and Steps” in the “Console Navigation” chapter of Getting Started); click **Next** or **Done** until you reach a screen containing a **Save** button. Then click **Save** and continue as needed until you return to the Main Menu.

   If your modification requires related configuration changes, PingFederate provides error messages indicating the necessary steps and then guides you to the related screens (unless you click **Cancel**).

**Important:** Be sure to click **Save** whenever that button appears, if you want to keep your changes.
Configuring IdP Auto-Connect

When your IdP partner is also using PingFederate 5 or higher (or is otherwise able to provide interoperable SAML 2.0 metadata via HTTP on demand), you may choose to use Auto-Connect for that partner (see “Using Auto-Connect” on page 30). This configuration can be shared among an unlimited number of SAML 2.0 partners.

**Note:** You enable the SAML 2.0 Auto-Connect profile under System Settings (see “Choosing Roles and Protocols” on page 106).

Once Auto-Connect is enabled on your PingFederate server, complete the configuration from the Main Menu under SP Configuration. This configuration entails:

- Setting up a common connection for all Auto-Connect partners
- Establishing a list of IdP partner domains authorized to use the connection

**Configuring the Initial Setup**

The basic configuration for IdP Auto-Connect requires only:

- Choosing a signing certificate for authentication requests and other SAML messages
- Configuring user-session creation information

All other partner-connection specifications are handled automatically at runtime.

**Choosing a Certificate**

For Auto-Connect runtime processing, authentication requests and SLO messages must be signed, since they are sent over either the POST or redirect bindings (see “SAML 2.0 Profiles” in the “Supported Standards” chapter of Getting Started).

**Note:** The signing certificate is embedded in your server’s Auto-Connect metadata (see “Using Auto-Connect” on page 30); there is no need to exchange certificates with your partners.

You can use the same certificate used for signing metadata (see “Configuring Auto-Connect Metadata Signing” on page 114). If you use a different certificate, ensure that it meets Auto-Connect validation requirements (see “Auto-Connect Security Model” on page 33).
To specify a certificate:

1. Select the certificate from the drop-down list.
   
   If you have not yet created or imported your certificate into PingFederate, click **Manage Certificates** (see “Digital Signing and Decryption Keys and Certificates” on page 209).

2. (Optional) Select the Signing Algorithm from the drop-down list.
   
   The default selection, RSA SHA1, is the most commonly used. Make a different selection only if you and your connection partner have agreed to use a stronger algorithm.

**Configuring User-Session Creation**

Configuring user-session creation for Auto-Connect is similar to configuring the same settings for regular partner connections.

► Click **Configure User-Session Creation** to continue.

For configuration information, refer to sections under “User-Session Creation” on page 367.

**Note:** Attributes sent from the IdP via Auto-Connect are passed to your applications, regardless of whether they are listed in the attribute contract (see “Defining an Attribute Contract” on page 369).

**Connection Activation and Summary**

When you finish configuring your IdP Auto-Connect initial setup, you may choose to activate the common connection immediately on the Activation & Summary screen. (No runtime processing occurs until your partner’s Auto-Connect gateway is also established and a user initiates an SSO or SLO event.)

**Important:** Regardless of whether you choose to activate a newly configured connection now or later, you must click **Save** on the Activation & Summary screen if you want to keep the configuration.

You can deactivate the connection at any time (for maintenance, for example). While a connection is inactive, all SSO or SLO transactions to or from Auto-Connect partners are disabled.

**To change a Connection Status:**

► Select Active or Inactive and then click **Save**.
To modify a setting:
1. Locate the currently configured setting under one of the summary headings and then click the subheading above the data.

   Note: Changes made to Auto-Connect settings will be out of sync, temporarily, with metadata caches that any currently active partners might be using. If your connection is in production, you might wish to lower your server’s metadata lifetime in advance of making configuration changes (see “Configuring Auto-Connect Metadata Lifetime” on page 115).

2. Change the information and click Save, if available.
   If Save is not available, additional, dependent changes are required; click Next or Done until you reach a screen containing a Save button. Then click Save and continue as needed until you return to the Main Menu.

Specifying Allowed IdP Domains

This screen provides PingFederate with a list of trusted domain names of your Auto-Connect partners.

Normally, when PingFederate receives an SSO request from a Web application at your site (see “/sp/startSSO.ping” on page 533), the runtime engine completes the connection automatically using metadata obtained from a standard, public location—http://saml.<domain_name>.(See “Using Auto-Connect” on page 30.) Alternatively, if an Auto-Connect partner elects not to use the standard location, you can supply the applicable URL.

To add a domain:
► Enter a Domain Name and click Add.

To specify a URL for metadata retrieval:
1. Click Advanced View.
2. Enter the Domain Name if you have not already done so.
3. Enter the Metadata Service URL.
   This entry must be obtained from your Auto-Connect partner.
4. Click Add.

   Note: Once you have added the URL, you cannot return to the Basic View unless you first remove the URL value using the procedure below.
To edit an entry:
1. Click Edit under Action for the entry.
2. Make your change and click Update.

To delete an entry:
   ▶ Click Delete under Action for the entry.
WS-Trust STS Configuration

The PingFederate WS-Trust STS provides security-token validation and creation to extend SSO access to identity-enabled Web Services (see “About WS-Trust STS” on page 6).

The chapter provides instructions for configuring the WS-Trust STS, including:

- “Server Settings"
- “IdP Configuration for STS”
- “SP Configuration for STS”

Server Settings

To use the PingFederate WS-Trust STS for partner connections, start by enabling the WS-Trust protocol under Server Settings on the Roles and Protocols screen (see “Choosing Roles and Protocols” on page 106). Once the protocol is enabled, you must identify the STS server with a unique federation identifier for both SAML 2.0 and SAML 1.1 tokens (unless these IDs are already established for corresponding browser-based SSO protocols).

In addition, also under Server Settings, you have the option of requiring authentication globally for access to STS endpoints—(see “Configuring STS Authentication” on page 438).

Enabling the WS-Trust STS

You can enable the WS-Trust STS when you first install PingFederate (see “Running PingFederate for the First Time” in the “Installation” chapter of Getting Started). If you have already installed PingFederate or are upgrading to a new version, use the following procedure.

To enable WS-Trust and make the STS available for partner connections:
1. On the Main Menu under System Settings, click **Server Settings**.
2. Click **Roles and Protocols** under the Server Settings tab.
3. Select WS-Trust for either the IdP or the SP role, or both, depending on your requirements.

**Note:** PingFederate fully supports the STS with or without selections of any of the Browser SSO protocols listed above the WS-Trust selections. SAML 1.1 and 2.0 token handling is independent of supported SSO protocols chosen here.

4. Click Next.

5. On the Federation Info screen, ensure all required fields are completed.

**Note:** Identifiers are needed for both SAML 2.0 and SAML 1.x to enable the STS to issue either type of token when requested. If you have not established a federation ID for either of these protocols or do not expect to use one or the other, enter a placeholder (in any format) and return later if needed. (For more information about the fields on this screen, see “Specifying Federation Information” on page 108).

6. (Optional) Click Next to go to the WS-Trust STS Settings screen (see the next section, “Configuring STS Authentication”).

7. Click Save (on any screen).

### Configuring STS Authentication

Server settings may be configured to require that client applications provide credentials to access the PingFederate STS. This is recommended for IdP configurations using the Username Token Processor (available separately).
For other token processors and token generators, trust in the identity of the client is conveyed within the token itself and verified as part of processing. However, administrators may want to add another layer of security by limiting access to only authenticated clients.

**Note:** You can configure STS authentication to either apply globally to all token formats and for all IdP and SP partner connections or token-to-token mappings or, using more fine grained controls, at the connection level via Issuance Criteria (see “STS Token Exchange Mapping” on page 143).

To continue, click **Configure WS-Trust STS Authentication**.

### Selecting Authentication Methods

You can choose either HTTP Basic or mutual SSL/TLS authentication (or both) on the Authentication Methods screen. (Note that if both methods are configured, all clients must authenticate using both, not one or the other.)

**Important:** If you choose mutual SSL/TLS authentication, you must configure a secondary PingFederate SSL port (see the property `pf.secondary.https.port` in the table under “Changing Configuration Parameters” on page 74).

**Note:** You can configure STS authentication to either apply globally to all token formats and for all IdP and SP partner connections or token-to-token mappings or, using more fine grained controls, at the connection level via Issuance Criteria (see “STS Token Exchange Mapping” on page 143).

#### Configuring Basic Authentication

For HTTP Basic authentication, create username/password pairs (“Users”) for all client applications needing access to the STS.
On the HTTP Basic Authentication screen, you can also delete users and update account passwords.

**To add users:**

1. Click **Create User**.
2. On the User Account screen, enter a Username and Password, and confirm the password.
   
   Passwords must be at least six characters long, containing at least one uppercase, one lowercase, and one numeric character.
3. Click **Done**.
4. Repeat the preceding steps as needed.
5. On the HTTP Basic Authentication screen, click **Next**.
   
   (If you are also configuring SSL authentication, complete that configuration and click **Next** to reach the Summary screen (see “Configuring Mutual SSL Authentication” on page 441).)
6. On the Summary screen, click **Done**.
7. On the WS-Trust STS Settings screen, click **Save**.

**To update an account password:**

1. Click the Username.
2. On the User Account screen, enter the Current User Password and a New Password, with confirmation.
   
   Passwords must be at least six characters long, containing at least one uppercase, one lowercase, and one numeric character.
3. Click **Done**.
4. On the HTTP Basic Authentication screen, click **Done**.
5. On the WS-Trust STS Settings screen, click **Save**.

**To delete a user:**

1. Click **Delete** under Action for the Username.
2. Click **Done** (or **Next** for new configuration).
3. Click **Save** when you reach the WS-Trust STS Settings screen.
Configuring Mutual SSL Authentication

When SSL authentication is selected on the Authentication Methods screen, the configuration begins on the Mutual SSL Authentication screen.

**Important:** If you choose mutual SSL/TLS authentication, you must configure a secondary PingFederate SSL port (see the property `pf.secondary.https.port` in the table under “Changing Configuration Parameters” on page 74)

To continue, click **Configure Mutual SSL Authentication**.

Choosing Certificate Authentication Options

On the Authentication Options screen, select whether to verify client SSL certificates against a list of Subject Distinguished Names (DNs) or a list of issuer public certificates imported into PingFederate.

**Note:** You can choose both options. However, note that they are not used alternatively at runtime; both validations are applied.

To continue, select one or both methods and click **Next**.

For information about restricting access by Subject DN, see the next section. For information about restricting access by certificate, see “Managing Allowed Issuer Certificates” on page 442.

Managing Allowed Subject DNs

On the Allowed Subject DNs screen you can add, edit, or delete Subject DNs for clients allowed to access the PingFederate STS.
**WS-Trust STS Configuration**

**To add DNs:**
1. Enter a valid Subject DN for a partner STS client and click **Add**.
2. Add other DNs as needed.
3. For a new configuration, click **Next** or **Done** to continue.
4. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

**To edit DNs:**
1. Click **Edit** under Action for the Subject DN.
2. Make changes and click **Update**.
3. Click **Done**.
4. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

**To delete entries:**
1. Click **Delete** under Action for the Subject DN.
2. Click **Done**.
3. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

**Managing Allowed Issuer Certificates**

When STS access is restricted by issuer certificate, the Allowed Issuer Certificates screen provides a means of maintaining a list of valid certificates.

**To add certificates:**
1. Select the certificate from the drop-down list and click **Add**.
   - If the certificate you are looking for is not in the list, click **Manage Certificates** to import it from your file system.
2. Add other certificates as needed.

On this screen you can add or remove certificates.
3. For a new configuration, click **Next** or **Done** to continue.

4. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

**To delete a certificate from the list:**
1. Click **Remove** under Action for the Issuer Certificate.
2. Click **Done**.
3. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

**Using the Mutual SSL Summary Screen**

When you have finished configuring Mutual SSL Authentication, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.

- To save a new or modified configuration, click **Done** on successive screens until you reach the WS-Trust STS Settings screen and then click **Save**.

**Using the STS Summary Screen**

When you have finished configuring WS-Trust STS Settings, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.

- If you are editing an existing connection, click **Done** and on the WS-Trust STS Settings screen click **Save**.

**IdP Configuration for STS**

This section covers the IdP configuration for the PingFederate WS-Trust STS, which involves:

- “Configuring Token Processors”
- “Managing STS Request Parameters” (Optional)
- “Configuring SP Connections for STS”

**Configuring Token Processors**

Token Processors are used to validate incoming tokens and token requests to the STS (see “Token Processors and Generators” on page 7). Token Processors for SAML, OAuth, JWT tokens, and username tokens are included with the PingFederate installation. This section provides guidance on configuring “instances” of these installed Token Processors.
You must configure at least one processor in order to set up an STS connection or token-to-token mapping (see “STS Token Exchange Mapping” on page 143).

**Important:** If more than one instance of the same token-processor type is configured for a connection or token-to-token mapping, clients calling either of the PingFederate STS endpoints must add a query parameter, TokenProcessorId, and specify the Instance Id. (For endpoint information, see “Viewing Protocol Endpoints” on page 245.)

For example:

```
https://<pf_host>:<pf_port>/idp/sts.wst
?TokenProcessorId=saml2firstinstance
```

Additional Token Processors may be downloaded from the Ping Identity Web site (www.pingidentity.com/support-and-downloads). For configuration information, please consult documentation provided for the respective add-on processor.

**To begin configuring Token Processors:**

- On the Main Menu, click **Token Processors** under Application Integration Settings for IdP Configuration.

If this link is not shown, ensure that the WS-Trust STS is enabled in **Server Settings** (see “Enabling the WS-Trust STS” on page 437).

<table>
<thead>
<tr>
<th>Main</th>
<th>Manage Token Processor Instances</th>
</tr>
</thead>
</table>

**PingFederate uses token processors to handle incoming security tokens. Here you can manage ‘instances’ of token processors that SP connections may use to fulfill Attribute Contracts sent to partners.**

**Table:**

<table>
<thead>
<tr>
<th>INSTANCE NAME</th>
<th>INSTANCE ID</th>
<th>TYPE</th>
<th>PARENT NAME</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>OAuth Bearer ATM Token Processor</td>
<td>OAuthBearerATMP</td>
<td>OAuth Bearer Access Token Token Processor</td>
<td></td>
<td>Delete</td>
</tr>
<tr>
<td>SAML 1.1</td>
<td>sam11</td>
<td>SAML 1.1 Token Processor</td>
<td>None Available - In Use</td>
<td>Delete</td>
</tr>
<tr>
<td>SAML 2.0</td>
<td>sam12</td>
<td>SAML 2.0 Token Processor</td>
<td>None Available - In Use</td>
<td>Delete</td>
</tr>
<tr>
<td>SAML 2.0 Token Processor Child</td>
<td>sam12child</td>
<td>SAML 2.0 Token Processor</td>
<td>Username Token</td>
<td>Delete</td>
</tr>
<tr>
<td>Username Token</td>
<td>UsernameTokenProcessor</td>
<td>Username Token Processor</td>
<td>Username Token</td>
<td>Delete</td>
</tr>
<tr>
<td>Username Token Child</td>
<td>UsernameTokenChild</td>
<td>Username Token Processor</td>
<td>Username Token</td>
<td>Delete</td>
</tr>
</tbody>
</table>

- **To configure a new token-processor instance:**
  - Click **Create New Instance**.

- **To edit an existing instance:**
  - Click the Instance Name and click the step you need to change.

- **To delete an instance:**
  1. Click **Delete** next to the Instance Name. (To undo the deletion, click **Undelete**.)
     
     **Note:** This option is available only if the processor instance is not in use for a connection.

  2. Click **Save** to confirm the deletion.
Selecting a Token Processor Type

The first step in creating a token-processor instance is choosing the processor type.

1. Enter the Instance Name and Instance Id on the Type screen.
2. Select the processor Type from the drop-down menu.
3. (Optional) Select a Parent Instance from the drop-down list.
   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
4. Click Next and enter information on the configuration screen for this token-processor instance.

   This configuration varies depending on the token processors deployed on your server. For add-on processors please consult the online documentation referenced in the download package, or look under Product Documentation at pingidentity.com.

   For token processors bundled with PingFederate, refer to one of the following sections:
   - “Configuring a SAML Token Processor Instance”, next
   - “Configuring an OAuth Token Processor Instance” on page 447
   - “Configuring a JSON Web Token (JWT) Processor Instance” on page 447
   - “Configuring a Username Token Processor Instance” on page 448

Configuring a SAML Token Processor Instance

On the Instance Configuration screen, you may use signing-certificate DN checking to limit the valid signatures and certificates for token requests accepted for this SAML token type. (By default, the STS validates digital signatures using all trusted Certificate Authorities (CAs) imported into PingFederate.)

At minimum on this screen, you must indicate a unique identifier for the PingFederate STS. To be accepted, an incoming SAML token must contain this ID in its <audience> element.
Chapter 8
WS-Trust STS Configuration

To configure the token-processor instance for certificate validation:

1. Enter a URI for Audience.
   This is the ID for the STS for either SAML 1.1 or SAML 2.0 tokens, depending on which processor you are configuring (see “Specifying Federation Information” on page 108).

2. (Optional) Click the Add a new... link under Action for either Valid Certificate Issuer DNs or Valid Certificate Subject DNs.
   You can use both lists.

   **Important:** When both types of validation are configured, then the certificate used to validate signatures must match an entry in both lists. If only Subject DNs are listed on this screen, then the certificate Issuer DN is not checked and its Subject DN must match one of the entries in the Subject DNs list. If only Issuer DNs are listed here, then the certificate Subject DN is not checked and its Issuer DN must match one of the entries in the Issuer DNs list. If neither Issuer DNs nor Subject DNs are listed, then all certificates are treated as valid for purposes of verification.

3. (Optional) Enter a Valid DN and click Update.

4. (Optional) Repeat the previous steps as needed to add more DNs.
Configuring an OAuth Token Processor Instance

The PingFederate STS provides validation for OAuth Bearer tokens (see “About OAuth” on page 10). Generally, a client would send the token in order to receive a SAML token in exchange.

**Note:** To use this token processor, you must first configure an access-token attribute contract (see “Access Token Management” on page 165).

### Configuring a JSON Web Token (JWT) Processor Instance

The PingFederate STS provides validation for JSON web tokens.

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JWKS Endpoint URI</td>
<td>The URI of the JWKS endpoint. A set of JSON Web Keys (JWK) are downloaded from this endpoint and used for JWT signature verification.</td>
</tr>
<tr>
<td>Issuer</td>
<td>A unique identifier for the issuer of the JWT.</td>
</tr>
<tr>
<td>Expiry Tolerance</td>
<td>The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.</td>
</tr>
</tbody>
</table>
Chapter 8
WS-Trust STS Configuration

To configure the token-processor instance for JWT validation:
1. Enter a JWKS Endpoint URI.
2. (Required) Enter the Issuer.
3. Enter an Expiry Tolerance.
4. Click Next.

Configuring a Username Token Processor Instance

The PingFederate Username Token Translator provides an Identity Provider (IdP) Token Processor for the PingFederate WS-Trust Security Token Service (STS). The Token Processor allows the STS to accept and validate a username security token from a Web Service Client (WSC) and then map user attributes into a SAML token for the WSC to send to a Web Service Provider (WSP).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issuer</td>
<td>(Required) A unique identifier for the issuer of the JWT.</td>
</tr>
<tr>
<td>Expiry Tolerance</td>
<td>The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.</td>
</tr>
</tbody>
</table>

To configure the token-processor instance for Username Token validation:
1. On the Instance Configuration screen, click Add a new row to ‘Password Credential Validators’ to define a credential-authentication mechanism instance for the adapter.
2. Select a password credential validator from the list and click Update.

Add as many validators as necessary. Use Move Up and Move Down to adjust

Note: If this is a child instance, select the override checkbox to modify the configuration.
the order in which you want PingFederate to attempt credential authentication. If the first mechanism fails to validate the credentials, PingFederate moves sequentially through the list until credential validation succeeds. If none of the defined password credential validators is able to authenticate the user's credentials, and the challenge retries maximum has been reached, the process fails.

**Note:** If usernames overlap across multiple password credential validators, the failovers could lockout those accounts in their source locations.

3. Click Next.

**Extending a Processor Contract**

Token processors allow administrators to add to a built-in list of user attributes that the processor returns from an incoming token—an extended processor-attribute contract.

To add an attribute:

**Note:** If this is a child instance, select the override checkbox to modify the configuration.

- (Optional) Enter the attribute name in the text box and click **Add**.

**Important:** For the OAuth 2.0 Bearer Token Processor, added attributes must also be among those configured under Access Token Management (see “Defining the Access Token Attribute Contract” on page 170).

**Setting Attribute Masking**

On the Token Attributes screen, you can choose to mask attribute values that PingFederate logs from this processor instance at runtime (see “Attribute Masking” on page 23).
Chapter 8
WS-Trust STS Configuration

To mask an attribute in log files:

Note: If this is a child instance, select the override checkbox to modify the configuration.

► (Optional) Under Mask Log Values select the attribute whose value you want to mask.
If OGNL expressions might be used to map derived values into outgoing tokens and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

Editing and Saving Processor Instances
From the Summary screen, you can reach processor settings for editing.

To edit the configuration:
1. Click the heading above the information you want to change.
2. Make your changes.
3. Click Done on the configuration page and Save on the Manage Token Processors screen.

To save a processor instance:
1. Click Done on the Summary screen.
2. Click Save on the Manage Token Processors screen.

Managing STS Request Parameters
As an option for configuring PingFederate to act as a WS-Trust STS, an administrator can define sets of RST metadata parameters that can be used to map attribute values into issued security tokens. After these request contracts are defined, you can make them available when configuring WS-Trust STS settings for SP-partner connections (see “Selecting a Request Contract” on page 455).
To reach this screen, click **STS Request Parameters** under Application Integration Settings on the Main Menu.

If this link is not present, WS-Trust is not enabled (see “Enabling the WS-Trust STS” on page 437).

To add a new set of request parameters, click **Add New Request Contract**.

To edit an existing contract, click its **Contract Name**.

### Creating a Request Contract

On the Create Request Contract screen, identify the contract and define parameters that will be available in token requests (as associated with this contract for partner connections—see “Managing STS Request Parameters” on page 450).

#### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract Name</td>
<td>A descriptive name for the Contract—for example, a Web Service Client or Provider.</td>
</tr>
<tr>
<td>Contract ID</td>
<td>An internal identifier—must be alphanumeric with no spaces.</td>
</tr>
<tr>
<td>Parameters to be provided in the request</td>
<td>A list of request parameters for this Contract (see instructions below).</td>
</tr>
</tbody>
</table>

**To add a Parameter:**

- Enter the Parameter Name in the text box and click **Add**.

**To modify a Parameter Name:**

1. Click **Edit** under Action for the Parameter Name.
2. Edit the name and click **Update**.

**Note:** If you change your mind, be sure to click the **Cancel** link in the Actions column, not the **Cancel** button, which discards any other changes you might have made.

**To delete a Parameter:**

- Click **Delete** for the Parameter Name.
Chapter 8
WS-Trust STS Configuration

Configuring SP Connections for STS

You can configure an STS connection to an SP partner either in conjunction with browser-based SSO or independently.

To enable STS for a new connection, or to add the capability to an existing connection:

1. Select the WS-Trust STS option on the Connection Type screen (see “Choosing a Connection Type” on page 252).

   Note: Before this option can be selected, the WS-Trust protocol must be enabled in Server Settings (see “Server Settings” on page 437)

2. Select a Default Token Type.
   The Default Token Type, either SAML 1.1 or 2.0, is used when a Web Service client does not specify in the token request what token type the STS should issue.

   Note: The Default Token Type does not need to match the Protocol indicated on the screen for SSO (when applicable).

When the option is enabled, the configuration starts on the WS-Trust STS screen.

To continue, click Configure WS-Trust STS.

Configuring IdP Protocol Settings

On this screen, use the Add button for the Partner Service Identifier field to enter one or more URLs for your partner’s Web Service(s). Each of these identifiers is compared to the element <AppliesTo> in Requests for Security Tokens (RSTs) and may be either a complete URL or a base URL for matching variable ports or paths.
Also on this screen, options are available for adding signature or encryption protection to outgoing SAML tokens and for enabling support of two additional token-type requests:

- For SAML 1.1 and SAML 2.0, you can choose to generate a symmetric key to be used in conjunction with the “Holder of Key” designation for the assertion’s Subject Confirmation Method (for information about HoK assertions, see, for example, “Web Services Security SAML Token Profile” (docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0.pdf).

- For SAML 2.0, you can choose to encrypt the assertion.

- Enabling the OAuth SAML Bearer Profile permits two additional token-type requests based on these OAuth grant types:
  - SAML 2.0 Bearer Assertion Grant Type
  - OAuth Access Token via SAML 2.0 Bearer Assertion Grant Type

See “STS OAuth Integration” on page 9 for more information on the use of these token-type requests.

**Note:** You can make any or all selections if you expect requests for these types of tokens. These selections are independent of the Default Token Type selected previously (see “Configuring SP Connections for STS” on page 452).

When you make one (or all) of these selections, you are asked to choose a signing or XML encryption certificate later in the connection setup, unless required certificates are already in place for an existing browser-based SSO connection. (PingFederate uses the same certificates to handle signing/encryption requirements for both Browser SSO and WS-Trust STS—for more information, see “Configuring Credentials” on page 306.)

**Setting a Token Lifetime**

Standards require a window of time during which a security token is considered valid. Each token has a time-stamp XML element as well as elements indicating the allowable lifetime of the token (in minutes) before and after the token time stamp.

<table>
<thead>
<tr>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Minutes Before</strong></td>
</tr>
<tr>
<td><strong>Minutes After</strong></td>
</tr>
</tbody>
</table>

**Field Descriptions**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minutes Before</td>
<td>The amount of time before the token was issued during which it is to be considered valid.</td>
</tr>
<tr>
<td>Minutes After</td>
<td>The amount of time after the token was issued during which it is to be considered valid.</td>
</tr>
</tbody>
</table>

To change the default times:

- (Optional) Edit the desired setting(s) and click **Next** or **Save**.
Chapter 8
WS-Trust STS Configuration

Configuring Token Creation

For the PingFederate STS to issue a security token in response to requests for partner services, you must indicate what user attributes are to be included in the token (the "attribute contract"). The attribute values sent in the token are then derived by mapping those available from the Token Processor you select (see “Fulfilling the Attribute Contract” on page 470). As with Browser SSO, the mapping can be augmented using local data stores, variable or constant text, or expressions.

Details of this configuration are handled under the Token Creation task.

To continue, click Configure Token Creation.

Defining an STS Attribute Contract

An attribute contract is the set of user attributes that a Web Service Client at your site expects to receive in security tokens issued for this connection (see “Attribute Contracts” on page 19). You identify these attributes on this screen.

(This screen presents an additional option when SAML 1.1 is chosen as the Default Token Type on the Connection Type screen.)

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click Attribute Contract on the Summary screen.
To add an attribute:

1. Enter the attribute name in the text box.
   Attribute names are case-sensitive and must correspond to the attribute names (including claims) expected by the requesting WSC.

   **Tip:** The Format attribute associated with the NameID element in outgoing SAML tokens may be set when needed by adding an attribute called `SAML_NAME_FORMAT`. The value of that attribute can then be mapped later (see “Fulfilling the Attribute Contract” on page 470).

   For information about the NameID elements and applicable URI values, locate the SAML 2.0 specification at oasis-open.org/specs.

2. (Optional) For SAML 1.1 tokens, select the Attribute Namespace.
   This field appears only when SAML 1.1 is chosen as the Default Token Type on the Connection Type screen (see “Configuring SP Connections for STS” on page 452).
   Change the default Namespace selection if you and your SP partner have agreed to a specific namespace (see “STS Namespaces” on page 21).

   **Tip:** You can add a special attribute, `SAML_AUTHN_CTX`, to indicate to the SP (if required) the type of credentials used to authenticate to the IdP application—authentication context. Map a value for the authentication context on the attribute-mapping screen later in the configuration, from any available attribute source, including the RST if a requested context is specified as a request parameter (see “Fulfilling the Attribute Contract” on page 470).

3. Click Add.

To modify an attribute name or namespace:

1. Click Edit under Action for the attribute.
2. Make the change and click Update.

To delete an attribute:

- Click Delete under Action for the attribute.

Selecting a Request Contract

This optional setting allows you to use XML parameters contained in RSTs for token-attribute mapping (see “Managing STS Request Parameters” on page 450).
If you are not using request parameters, click **Next** to continue.

To use request parameters, select the checkbox and choose a Request Contract from the drop-down list.

If the contract you want is not shown, click **Manage STS Request Parameters**.

When you choose a contract, you will enable an option to select Request from the drop-down **Source** list on the attribute-mapping screen (see “Fulfilling the Attribute Contract” on page 470).

### IdP Token Processor Mapping

IdP token processors are responsible for validating incoming security tokens as part of an STS operation (see “Token Processors and Generators” on page 7). A configured and deployed token processor in PingFederate is known as a token processor instance. The same instance may be mapped by multiple connections.

Map one or more IdP token processor instances into each SP connection to handle all the token types that may be received from Web Service Clients associated with this SP partner.

When token processors are restricted to certain virtual server IDs, the allowed IDs are displayed under the **Virtual Server IDs** column.

You begin this configuration on the IdP Token Processor Mapping screen, where you choose to map instances of IdP token processors. If you have not yet configured an instance of the token processor you intend to use within this SP connection, see “Configuring Token Processors” on page 443.

#### To reach this screen for editing:

1. Click the connection name on the **Main Menu**.
   
   Click **Manage All SP**, if needed, to see a full list of connections.

2. Click **WS-Trust STS** under the **SP Connection** tab.

3. Click **Configure WS-Trust STS**.
4. Click **Token Creation** under the WS-Trust STS tab.
5. Click **Configure Token Creation**
6. Click **IdP Token Processor Mapping** on the Summary screen.

**To modify an existing Token Processor Instance:**

- Click its Name link.

**To begin configuring an Token Processor Instance for this connection:**

- Click **Map New Token Processor Instance**.

**Selecting a Token Processor Instance**

On this screen for a new connection, choose an instance of the Token Processor needed for this connection (see “Token Processors and Generators” on page 7).

You will use attributes returned from the token processor (the token-processor contract) to fulfill the attribute contract required for this partner and/or use them to look up additional attributes in a user-data store. You make this choice on the next screen (see “Retrieving Additional Attributes” on page 485).

- Choose a **Token Processor Instance** from the drop-down list and click **Next** to continue.

  (Optional) To override any token processor instance, select the **Override Instance Settings** checkbox (see “Overriding Token Processor Instances” on page 458).

  To create or change a processor instance, as needed, click **Manage Token Processor Instances**.

**To reach this screen for editing:**

1. Click the connection name on the **Main Menu**.
   - Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the **SP Connection** tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Creation** under the WS-Trust STS tab.
5. Click **Configure Token Creation**
6. Click **IdP Token Processor Mapping** on the Summary screen.
7. Click the Token Processor Instance Name.
Overriding Token Processor Instances

Overrides at the connection level simplify token management by allowing you to create a small set of token instances that define the base configuration requirements and then overriding settings at the connection level as needed.

You may override any token processor settings at the connection level either during or after connection mapping.

Alternatively, you can override any token processor instance to apply to several connections, see “Hierarchical Plug-in Configurations” on page 17 for more information about token processor overrides.

Note: Any changes to the base token processor instance are propagated to a connection provided the same changes are not overridden for the connection.

Click Override Instance Settings.

On each of the settings screens, make your changes, and then click Next. When you are finished, click Done to continue with token mapping.

Note: Override token-setting screens are functionally identical to those used for creating a new token connection. Refer to the table below to find sections in this manual containing configuration information and procedures.

The display of some screens listed in the table depends on the type of token you are configuring.

For information about the override token-setting screens, depending on the type of setting, use the following table:

<table>
<thead>
<tr>
<th>Override Screen</th>
<th>Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Configuration</td>
<td>See “Configuring Token Processors” on page 443.</td>
</tr>
<tr>
<td>Extended Contract</td>
<td>See “Extending a Processor Contract” on page 449.</td>
</tr>
<tr>
<td>Token Attributes</td>
<td>See “Setting Attribute Masking” on page 449.</td>
</tr>
</tbody>
</table>

To remove any token processor connection override, clear the Override Instance Settings checkbox, and then complete the rest of the setup.

Restricting a Token Processor to certain Virtual Server IDs

When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), you have the option to enforce
authentication requirements by restricting a token processor to certain virtual server IDs. This optional setting can be applied to each token processor added to the connection using virtual server IDs. By default, no restriction is imposed.

**To restrict a token processor to a subset of the available virtual server IDs:**

1. Select the **Restrict Virtual Server IDs** checkbox.
2. In the **Allowed Virtual Server IDs** area, select one or more virtual server IDs that you want to allow for this token processor.
3. Click **Next**.

**To reach this screen for editing:**

1. Click the connection name on the Main Menu.
   
   Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the SP Connection tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Creation** under the WS-Trust STS tab.
5. Click **Configure Token Creation**
6. Click **IdP Token Processor Mapping** on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click **Virtual Server IDs** on the Summary screen.

**Note:** The Virtual Server IDs screen is only available for connections using at least one virtual server ID, see “Federation Server Identification” on page 37.

**Retrieving Attributes**

For token creation, you can query local user-data stores to help fulfill the attribute contract, in conjunction with attribute values supplied by the token processor you are using with PingFederate (see “Configuring Token Processors” on page 443).

The values supplied by the token processor are shown under Token Processor Contract on the Attribute Retrieval screen.
Chapter 8
WS-Trust STS Configuration

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the **SP Connection** tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Creation** under the **WS-Trust STS** tab.
5. Click **Configure Token Creation**.
6. Click **IdP Token Processor Mapping** on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click **Attribute Retrieval** on the Summary screen.
   - If you choose to “Retrieve additional attributes . . .”, then you identify data stores and specify lookup queries next (see the next section “Configuring STS Attribute Sources and User Lookup” on page 460).
   - If you “Use only the Token Processor Contract values . . .”, then you map values for the attribute contract next (see “Fulfilling the Attribute Contract” on page 470).

**Tip:** To determine whether you need to look up additional values, compare the token-processor contract against the attribute contract (see “Defining an STS Attribute Contract” on page 454). If the attribute contract requires more information, determine whether a local data store can supply it. (You can also choose to use text constants or expressions for certain information—see “Fulfilling the Attribute Contract” on page 470.)

**Configuring STS Attribute Sources and User Lookup**

Attribute sources are specific database, directory, or custom data store locations containing information that may be needed for the attribute contract (see “Defining an STS Attribute Contract” on page 454). Attribute sources can be reused across connections to other SP partners.

This portion of the connection configuration allows you to configure one or more data stores to look up attributes and to set up search parameters.

**Note:** Queries are executed in the order of Attribute Sources shown. Use the move up/move down controls as needed to adjust the order.
To configure an attribute source:

- Click Add Attribute Source and complete the setup steps (see “Configuring a Data Store for STS”, next).

To modify an attribute source configuration:

1. Click the attribute source Description link.
2. Click Save on the screen you change.

**Note:** Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click Save or Done when either of those options appears.

To reach this screen for editing:

1. Click the connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click IdP Token Processor Mapping on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click Attribute Source & User Lookup under the IdP Token Processor Mapping tab.

   If this step is not listed, then this instance is configured to use token-processor values only (see “Retrieving Attributes” on page 459).

Configuring a Data Store for STS

This screen allows you to choose a data store from a previously configured list (see “Managing Data Stores” on page 116). Attribute values extracted from one or more data stores are used to help fulfill the attribute contract (see “Defining an STS Attribute Contract” on page 454).
To reach this screen for editing:

1. Click the connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click IdP Token Processor Mapping on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click Attribute Source & User Lookup under the IdP Token Processor Mapping tab.
   - If this step is not listed, then this instance is configured to use token-processor values only (see “Retrieving Attributes” on page 459).
9. Click the attribute source Description link.

To define an attribute source:

1. Enter an Attribute Source Id to uniquely identify the data source for the mapping.
2. Use Attribute Source Description to specify an attribute source name that distinguishes this user lookup for the selected data store.

**Note:** PingFederate appends this description to the data store type in the Source list on the Attribute Contract Fulfillment screen (see “Fulfilling the Attribute Contract” on page 470).

3. Choose an Active Data Store and click Next.
   - A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click Manage Data Stores to add it (see “Managing Data Stores” on page 116).

Setting Up the Attribute Source

See the following sections in this manual, depending on the type of data store:
Selecting an STS JDBC Database Table and Columns

When you choose to use a database source for attributes, you follow this path through the configuration steps.

On this screen you begin to specify exactly where additional data can be found to complete the attribute contract when you send a security token to this SP (see “Defining an STS Attribute Contract” on page 454). Only one table may be used as a source of data for a JDBC lookup.

Important: (For MySQL users) To allow for table and column names that may contain spaces, PingFederate inserts double quotes around the names at runtime. To avoid SQL syntax errors resulting from the quotes, add the session variable `sql_mode=ANSI_QUOTES` to the connection string. For example:

```
```

For more information, see:

- [dev.mysql.com/doc/refman/5.0/en/identifiers.html](http://dev.mysql.com/doc/refman/5.0/en/identifiers.html)
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Lists the table structure that stores information within a database. Some databases, such as Oracle, require selection of a specific schema for a JDBC query. Other databases, such as MySQL, do not require selection of a schema.</td>
</tr>
<tr>
<td>Table</td>
<td>The name of the table contained in the database. Use the drop-down to change the table.</td>
</tr>
<tr>
<td>Columns to return from SELECT</td>
<td>Displays selected table columns. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:

1. Click the connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click IdP Token Processor Mapping on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click Database Table and Columns under the IdP Token Processor Mapping tab.

To select a database table and columns for queries:

1. Choose a Schema file (when applicable) from the drop-down list.
2. Choose a Table from the drop-down list.
3. Choose a name under Columns to Return from Select and click Add Attribute.

Tip: Click Refresh if you are updating an existing configuration and changes may have been made to the database.

Repeat this step for other columns as needed.

Note: You do not need to add a column here for it to be used as part of a search filter (see “Configuring a Database Query” next). Add only attributes from which you need actual values to pass in a token.
Configuring an STS Database Filter

The JDBC WHERE clause in PingFederate queries the data table you selected to retrieve a record associated with a particular value (or values) from the incoming security token. The clause is in the form:

```
WHERE column1=value1 [AND column2=value2] [OR ...]
```

The left side of the first variable pair uses a column name in the database table you selected (see “Selecting an STS JDBC Database Table and Columns” on page 463).

The right side generally uses values passed in from a token processor (variables, including the correct formatting, are listed under Token Processor Values—see “Configuring Token Processors” on page 443).

You can also apply additional search criteria from your own database, using any other columns from the targeted table.

**Tip:** To determine what attributes to look up during a query, click the View Attribute Contract link to see what information must be collected (see “Defining an STS Attribute Contract” on page 454). Then determine what information is coming in from the token processor (see “Retrieving Attributes” on page 459). Information not contained in the token-processor contract may be pulled from the data store look-up query.

**Note:** If you are retrieving attributes from multiple data stores using one mapping, attributes available from other sources, if previously configured, are listed near the bottom of the screen. For more information on multiple data-store mapping, see “Multiple Data Source Attribute Mapping” on page 22.

For more information about WHERE clauses, consult your DBMS documentation.

**EXAMPLE:**
```
userid='${username}'
```

In this example userid is the name of a column in the JDBC data store. On the right side, `${username}` returns the value of the username variable from the IdP token processor.

**Important:** You must use the `${}` syntax to retrieve the value of the enclosed variable and use single quotation marks around the `${}` characters.
Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where</td>
<td>WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: WHERE email='<a href="mailto:clive@company.com">clive@company.com</a>'.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   - Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click IdP Token Processor Mapping on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click Database Filter under the IdP Token Processor Mapping tab.

To construct the WHERE clause:
1. Enter the statement in the space provided, following the guidelines and example above.
   - The initial WHERE is optional.
2. Ensure the syntax and variable names are correct.
   - When you click Next, you will map attribute values returned from the database into the security token (see “Fulfilling the Attribute Contract” on page 470).

Configuring an LDAP Search

When you choose to use an LDAP source for attributes, you follow this path through the configuration steps.

On this screen you specify the branch of your LDAP hierarchy where you want PingFederate to look up user data.
IdP Configuration for STS

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. This field is optional if records are located at the LDAP root.</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Determines the node depth of the query. Select Subtree, One level or Object.</td>
</tr>
<tr>
<td>Root Object Class</td>
<td>The class containing the attributes you want.</td>
</tr>
<tr>
<td>Attributes to return from search</td>
<td>A list of attributes added from the drop-down list below. Subject DN is a default attribute, which may be used as the primary user identifier.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All SP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the SP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Creation under the WS-Trust STS tab.
5. Click Configure Token Creation
6. Click IdP Token Processor Mapping on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click LDAP Directory Search under the IdP Token Processor Mapping tab.

To select LDAP attributes:
1. (Optional) Enter a Base DN.
2. Select a Search Scope.
3. Select a Root Object Class.
4. Under Attributes to return from search, choose an attribute and click Add Attribute.
   Note that the attribute Subject DN is always returned by default.
5. Repeat the last step for other attributes as needed.
6. (Optional) Change the Search Scope or the Root Object Class if you want attributes from other locations.

**Note:** You do not need to add an attribute here for it to be used in a search filter (see "Configuring an LDAP Filter for STS"). Add only attributes from which you need actual values to pass into the outgoing security token.

### Specifying Encoding for Binary Directory Attributes for STS

The LDAP Binary Attribute Encoding Types screen appears when a binary attribute is added on the LDAP Directory Search screen. Because binary attribute data cannot be used in an assertion, use this screen to specify which encoding type (Base64 or Hex) you want to apply during attribute contract fulfillment.

For example, Microsoft® Office 365 uses objectGUID, an immutable Active Directory binary attribute associated with user accounts. Office 365 requires this binary data to be Base64-encoded to correlate provisioned federated user data to Active Directory accounts.

**Note:** Attributes are specified as binary when configuring an LDAP connection (see "Specifying LDAP Binary Attributes" on page 126).

![](image)

#### To select an attribute encoding type:

- Select the type of attribute encoding you want to apply for each attribute and click **Next**.

#### To reach this screen for editing:

1. Click the connection name on the **Main Menu**.
   
   Click **Manage All SP**, if needed, to see a full list of connections.

2. Click **WS-Trust STS** under the SP Connection tab.

3. Click **Configure WS-Trust STS**.

4. Click **Token Creation** under the WS-Trust STS tab.

5. Click **Configure Token Creation**

6. Click **IdP Token Processor Mapping** on the Summary screen.

7. Click the Token Processor Instance Name.

8. Click **LDAP Binary Attribute Encoding Types** under the IdP Token Processor Mapping tab.
Configuring an LDAP Filter for STS

The LDAP filter queries the data you selected to retrieve a record associated with a particular value (or values) from the incoming token. The filter is in the form:

\[ \text{attribute} = \${\text{value}} \]

The left-side variable is an attribute you selected earlier (see “Configuring an LDAP Search” on page 466).

The right side generally uses values passed in from the security token (variables, including the correct syntax, are listed under Security Token Values—see “Configuring Token Processors” on page 443).

You can also apply additional search criteria from your data store, using any other attributes from the targeted object classes.

Tip: Click “View List of Available LDAP Attributes” for a list from which you can copy and paste.

For general information about search filters, consult your LDAP documentation.

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>Narrows a search to locate requested data by either including or excluding specific records. An LDAP filter includes the attributes in the search and the value or range of values that the search is attempting to match. Searches are conducted by using three components: 1) at least one attribute (attribute data type) to search on, 2) a search filter operator that will determine what to match, and 3) the value of the attribute being sought. Searches must have at least one of each of these three components.</td>
</tr>
</tbody>
</table>

To reach this screen for editing:

1. Click the connection name on the Main Menu.
   
   Click Manage All SP, if needed, to see a full list of connections.

2. Click WS-Trust STS under the SP Connection tab.

3. Click Configure WS-Trust STS.
4. Click **Token Creation** under the WS-Trust STS tab.
5. Click **Configure Token Creation**
6. Click **IdP Token Processor Mapping** on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click **LDAP Filter** under the IdP Token Processor Mapping tab.

**To construct the LDAP filter:**

1. Enter the statement in the space provided, following the guidelines and example above.

   ![Note:](image) If you used an anonymous binding to create this LDAP connection, your access might be restricted (see “Configuring an LDAP Connection” on page 121).

2. Ensure the syntax and variable names are correct.
3. Click **Next**.

**Configuring STS Custom Source Filters**

When you choose to use a custom source for attributes, you follow this path through the configuration steps.

On this screen you specify a filter, or lookup query, for your custom data source. This screen display and the syntax of the filter depends on your developer's implementation of the custom source SDK.

**Selecting Custom STS Source Fields**

On the Configure Custom Source Fields screen, you can choose from among the fields shown to map to the attribute contract. These choices are supplied by the driver implementation. Select only those needed to fulfill the attribute contract for this partner connection.

**Fulfilling the Attribute Contract**

You map attributes for outgoing security tokens for this partner on the Attribute Contract Fulfillment screen.

<table>
<thead>
<tr>
<th>Attribute Contract</th>
<th>Source</th>
<th>Value</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAML_SUBJECT</td>
<td>- SELECT -</td>
<td></td>
<td>None available</td>
</tr>
</tbody>
</table>

Map each attribute to fulfill the Attribute Contract from one of these Sources:

- **Token**
  
  When you make this selection, the associated Value drop-down list is populated by the token processor.
• LDAP/JDBC/Custom

**Note:** PingFederate appends a description in parentheses for configured data store lookups (see “Configuring STS Attribute Sources and User Lookup” on page 460).

Values are returned from your attribute source (if you are using data store—see “Retrieving Attributes” on page 459). When you make this selection, the Value list is populated by the LDAP, JDBC, or Custom attributes you identified as an Attribute Source (see “Configuring an LDAP Search” on page 466, “Selecting an STS JDBC Database Table and Columns” on page 463, or “Configuring STS Custom Source Filters” on page 470).

• Context

Values are returned from the context of the transaction at runtime.

**Note:** The HTTP Request and STS SSL Client Certificate Chain selections are retrieved as Java objects rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

**Note:** When using the STS Basic Authentication Username, STS SSL Client Certificate’s Subject DN, or STS SSL Client Certificate Chain attributes, ensure the associated authentication is enabled and configured in WS-Trust STS Settings (see “Selecting Authentication Methods” on page 439).

• Request

Values are supplied from parameters in the token request received from the Web Service Client. This selection is available only if a Request Contract was selected earlier (see “Selecting a Request Contract” on page 455).

• Expression (when enabled)

This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

**Tip:** You can use an expression to insert an OAuth access token into the assertion for SP-partner use in OAuth transactions (see “About OAuth” on page 10). For information about this feature and a sample expression, refer to the PingFederate SDK Javadoc entry for the class com.pingidentity.sdk.oauth20.AccessTokenIssuer. Javadocs are located in the PingFederate installation: `<pf_install>/pingfederate/sdk/doc/index.html`. 
- **Text**
  
  The value is what you enter. This can be text only, or you can mix text with references to any of the values from the incoming token, using the `${attribute}` syntax.
  
  You can also enter values from your data store, when applicable, using this syntax:
  
  `${ds.attr-source-id.attribute}`
  
  where `attr-source-id` is the Attribute Source Id value (see “Configuring STS Attribute Sources and User Lookup” on page 460) and `attribute` is any of the data store attributes you select.

**To reach this screen for editing:**

1. Click the connection name on the Main Menu.
   
   Click Manage All SP, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the SP Connection tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Creation** under the WS-Trust STS tab.
5. Click **Configure Token Creation**.
6. Click **IdP Token Processor Mapping** on the Summary screen.
7. Click the Token Processor Instance Name.
8. Click **Attribute Contract Fulfillment** under the IdP Token Processor Mapping tab.

**To map attributes:**

1. Choose a Source for each Target attribute.
2. Choose (or enter) a Value for each Attribute.
   
   See “Map each attribute to fulfill the Attribute Contract from one of these Sources?” above. All values must be mapped.
3. Click **Next**.

### Selecting Issuance Criteria (Optional)

Use this screen to define criteria PingFederate can evaluate to determine whether to issue a SAML token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access identity-enabled Web services.

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
   
   Associated attributes appear in the Attribute Name drop-down list:
IdP Configuration for STS

- Context – Select to use values returned from the context of the transaction at runtime.

**Note:** The HTTP Request and STS SSL Client Certificate Chain selections are retrieved as Java objects rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

- LDAP/JDBC/Custom (if a data store is configured) – Select to access attributes returned from a data source.

**Note:** PingFederate appends a description in parentheses for configured data store lookups (see “Configuring STS Attribute Sources and User Lookup” on page 460).

- Mapped Attributes – Select to access the required attributes from the adapter.
- Request – Select to access parameters from a request contract.
- Token – Select to access the required attributes from the attribute contract.

2. Select an attribute name.

**Note:** When using the STS Basic Authentication Username, STS SSL Client Certificate’s Subject DN, or STS SSL Client Certificate Chain attributes, ensure the associated authentication is enabled and configured in WS-Trust STS Settings (see “Selecting Authentication Methods” on page 439).

3. Select the Condition you want to apply.

**Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

**Important:** When using the STS SSL Client Certificate’s Subject DN attribute, you must select one of the following conditions: equal to DN, not equal to DN, multi-value contains DN, or multi-value does not contain DN. These operators normalize the DN before comparison to accommodate for different string representations that are still considered equivalent (for example, case sensitivity, or whitespace).

4. Enter a value for the attribute.
5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

Errors result in a SOAP message returned. The Error Result field is used by the faultstring element for SOAP 1.1 and the Reason/Text element for SOAP 1.2.


Note: Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click Add.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

Note: All criteria must pass in order for a user to be authorized.

8. (Optional) Click Show Advanced Criteria to configure more complex evaluations using OGNL expressions.

Note: Expressions must be enabled for the Show Advanced Criteria button to appear (see “Enabling and Disabling Expressions” on page 573).

Important: When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

- Use the in-line editor box to enter the OGNL expression.
  
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.

- Use the Error Result box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

Note: If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click Add. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**
- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**
- Click **Delete** under Actions for the criteria and then click **Save**.

**Using the Mapping Summary Screen**

When you have finished configuring IdP Token Processor Mapping, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.

- If you are editing an existing connection, click **Done** on successive screens until you reach the WS-Trust STS screen, and then click **Save**.

**To save a new configuration:**
1. Click **Done** to return to the IdP Token Processor Mapping screen.
2. Click **Next** to go to the Token Creation Summary screen, and then click **Done**.
3. On the Token Creation screen, click **Done**.
4. On the WS-Trust STS screen, click **Save**.

**Request Error Handling**

If you are using request parameters to fulfill the attribute contract and the parameter values are not supplied, you can choose whether to continue or abort the token-creation process.

![Image](image-url)

**Note:** The Error Handling screen is presented only if a Request Contract is used for this configuration (see “Selecting a Request Contract” on page 455).

**Using the Token Creation Summary Screen**

When you have finished configuring Token Creation, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.
If you are editing an existing connection, click **Done** on successive screens until you reach the WS-Trust STS screen, and then click **Save**.

On the WS-Trust STS Summary screen, you can review the configuration for this connection.

If you need to make any changes to a new or existing connection, click the heading over the information you want to modify.
Chapter 9
WS-Trust STS Configuration

SP Configuration for STS

This section covers the SP configuration for STS, including:
- “Configuring Token Generators”
- “Configuring IdP Connections for STS”

Configuring Token Generators

Token Generators are used to issue security tokens that can be consumed by Web Services at your site (see “Token Processors and Generators” on page 7). Token Generators for SAML 2.0 and SAML 1.1 tokens are included with the PingFederate installation. This section provides guidance on configuring “instances” of either of the SAML Token Generators. You must configure at least one generator in order to set up an STS connection or token-to-token mapping (see “STS Token Exchange Mapping” on page 143).

Important: If more than one instance of the same token-generator type is configured for a connection or token-to-token mapping, clients calling either of the PingFederate STS endpoints must add a query parameter, TokenGeneratorId, and specify the Instance Id. (For endpoint information, see “Viewing SP Protocol Endpoints” on page 351.)

For example:
https://<pf_host>:<pf_port>/idp/sts.wst
?TokenGeneratorId=saml2firstinstance

Additional Token Generators may be downloaded from the Ping Identity Web site (www.pingidentity.com/support-and-downloads). For configuration information, please consult documentation provided for the respective add-on generator.

To begin configuring SAML 1.1 or 2.0 Token Generators:

On the Main Menu, click Token Generators under Application Integration Settings for SP Configuration.

If this link is not shown, ensure that the WS-Trust STS is enabled in Server Settings (see “Enabling the WS-Trust STS” on page 437).
To configure a new token-generator instance:
- Click Create New Instance

To edit an existing instance:
- Click the Instance Name and click the step you need to change.

To delete an instance:
1. Click Delete next to the Instance Name. (To undo the deletion, click Undelete.)
   
   **Note:** This option is available only if the generator instance is not in use for a connection.

2. Click Save to confirm the deletion.

**Selecting a Token Generator Type**

The first step in creating a SAML token-generator instance is choosing the generator type.

**To define an instance:**
1. Enter the Instance Name and Instance Id on the Type screen.
2. Select SAML 1.1 Token Generator or SAML 2.0 Token Generator from the drop-down menu.
3. (Optional) Select a Parent Instance from the drop-down list.
   
   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

4. Click Next.

**Configuring a Token Generator Instance**

On the Instance Configuration screen, you specify parameters for generated SAML tokens.
### Field Instructions

<table>
<thead>
<tr>
<th>Field</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minutes Before</td>
<td>Enter a numerical value. This element in a SAML token allows for any server clock variability.</td>
</tr>
<tr>
<td>Minutes After</td>
<td>Enter a numerical value. This element in a SAML token allows for any server clock variability.</td>
</tr>
<tr>
<td>Issuer</td>
<td>Enter the SAML 2.0 Entity ID or SAML 1.x Issuer specified on the Federation Information screen in Server Settings (see “Specifying Federation Information” on page 108).</td>
</tr>
<tr>
<td>Signing Certificate</td>
<td>Responses containing SAML tokens must be signed. If the signing certificate you need is not in the drop-down list, click Manage Signing Certificates near the bottom of the screen.</td>
</tr>
<tr>
<td>Include Certificate in KeyInfo</td>
<td>If selected, the entire public certificate is included with the assertion. Otherwise, a short hash reference to the certificate is sent instead.</td>
</tr>
<tr>
<td>Include Raw Key in KeyValue</td>
<td>If selected, the raw key is included in the &lt;KeyInfo&gt; element as well.</td>
</tr>
</tbody>
</table>
Token generators allow administrators to add to a built-in list of user attributes that the generator includes in the outgoing token—an extended generator-attribute contract.

To add an attribute:

- Enter the attribute name in the text box and click Add.

**Extending a Generator Contract**

To edit the configuration:
1. Click the heading above the information you want to change.
2. Make your changes.
3. Click Done on the configuration page and Save on the Manage Token Generators screen.

To save a generator instance:
1. Click Done on the Summary screen.
2. Click **Save** on the Manage Token Generators screen.

### Configuring IdP Connections for STS

You can configure an STS connection to an IdP partner either in conjunction with browser-based SSO or independently.

**To enable STS for a new connection, or to add the capability to an existing connection:**

- Select the WS-Trust STS option on the Connection Type screen (see “Choosing an IdP Connection Type” on page 358).

**Note:** Before this option can be selected, the WS-Trust protocol must be enabled in Server Settings (see “Server Settings” on page 437)

When the option is enabled, the configuration starts on the WS-Trust STS screen.

To continue, click **Configure WS-Trust STS**.

### Configuring STS Protocol Settings

On the Protocol Settings screen, choose whether to validate incoming SAML tokens or to validate and then also generate different tokens to enable SSO access to Web Services at your site.

Also on this screen, if incoming SAML 2.0 tokens for this connection are required to be encrypted, select the checkbox for decrypting assertions. When you make this selection, you will be required to choose a decryption certificate for this partner later in the connection configuration (if one is not already selected for Browser SSO purposes—see “Choosing a Decryption Key” on page 432).

- If you choose not to generate new tokens, then no further settings are needed for this task—click **Next** and refer to “Using the Token Generation Summary Screen” on page 499 for instructions on saving this configuration.
You will be asked later to choose a certificate with which to verify the signature on the incoming SAML token (see “Configuring Signature Verification Settings” on page 310).

# Configuring Token Generation

For the PingFederate STS to issue a security token that meets identity requirements of Web Services at your site, you must indicate what user attributes are included in the incoming token (the “attribute contract”). The attribute values from the incoming token can be then mapped to attributes in the token generator you select (see “Mapping Token Attributes” on page 494). As with Browser SSO, the mapping can be augmented using local data stores, variable or constant text, or expressions.

Details of this configuration are handled under the Token Generation task.

To continue, click **Configure Token Creation**.

## Specifying an Attribute Contract

An attribute contract is the set of user attributes expected in incoming security tokens (see “Attribute Contracts” on page 19). You identify these attributes on this screen.

Optionally, you can mask the values of attributes (other than `SAML_SUBJECT`) in the log files that PingFederate writes when it receives security tokens (see “Attribute Masking” on page 23).

To reach this screen for editing:

1. Click the connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the IdP Connection tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Generation** under the WS-Trust STS tab.
   
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).

5. Click **Configure Token Generation**

6. Click **Attribute Contract** on the Summary screen.

   **To add an attribute:**
   
   1. Enter the attribute name in the text box.
      
      Attribute names are case-sensitive and must correspond to the attribute names expected by the requester.
   
   2. (Optional) Select the checkbox under **Mask Values in Log**.
   
   3. Click **Add**.

   **To modify an attribute name:**
   
   1. Click **Edit** under **Action** for the attribute.
   
   2. Make the change and click **Update**.

   **To delete an attribute:**
   
   ▶ Click **Delete** under **Action** for the attribute.

**Mapping Token Generators**

Token generators provide a mechanism through which PingFederate can generate a local token based upon an incoming SAML token, including mapping user attributes to be included in the generated token. A configured and deployed token generator in PingFederate is known as a token-generator instance.

You can map one or more token generator instances into each IdP connection to satisfy multiple session-management requirements where needed. The same instances may be mapped by multiple connections.

When token generators are restricted to certain virtual server IDs, the allowed IDs are displayed under the Virtual Server IDs column.

The configuration begins on the **Token Generator Mapping & User Lookup** screen. If you have not yet configured an instance of a token generator you need for this connection, see “Configuring Token Generators” on page 476.
To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see "Configuring STS Protocol Settings" on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.

To modify an existing Token Generator Instance:
- Click its Name link.

To begin configuring an Token Generator Instance for this connection:
- Click Map New Token Generator Instance.

Selecting a Token Generator Instance

On this screen for a new connection, choose an instance of the Token Generator needed for this connection (see “Token Processors and Generators” on page 7).

You will use attributes contained in the incoming security token to fulfill the token generator contract for this STS connection and/or use them to look up additional attributes in a user-data store. You make this choice on the next screen (see “Retrieving Attributes” on page 459).

On this screen, choose a Token Generator Instance from the drop-down list and click Next to continue.

(Optional) To override any token generator instance, select the Override Instance Settings checkbox (see “Overriding Token Generator Instances” on page 483).

To create or change a Token Generator Instance, as needed, click Manage Token Generator Instances.

Overriding Token Generator Instances

Overrides at the connection level simplify token management by allowing you to create a small set of token instances that define the base configuration requirements and then overriding settings at the connection level as needed.

You may override any token generator settings at the connection level either during or after connection mapping.
Alternatively, you can override any token generator instance to apply to several connections, see “Hierarchical Plug-in Configurations” on page 17 for more information about token generator overrides.

**Note:** Any changes to the base token generator instance are propagated to a connection provided the same changes are not overridden for the connection.

- **Click Override Instance Settings.**
  
  On each of the settings screens, make your changes, and then click Next. When you are finished, click Done to continue with token mapping.

  **Note:** Override token-setting screens are functionally identical to those used for creating a new token connection. Refer to the table below to find sections in this manual containing configuration information and procedures.

  The display of some screens listed in the table depends on the type of token you are configuring.

  For information about the override token-setting screens, depending on the type of setting, use the following table:

<table>
<thead>
<tr>
<th>Override Screen</th>
<th>Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance Configuration</td>
<td>See “Configuring a Token Generator Instance” on page 477.</td>
</tr>
<tr>
<td>Extended Contract</td>
<td>See “Extending a Generator Contract” on page 479.</td>
</tr>
</tbody>
</table>

- **To remove any token generator connection override, clear the Override Instance Settings checkbox, and then complete the rest of the setup.**

**Restricting a Token Generator to certain Virtual Server IDs**

When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), you have the option to enforce authentication requirements by restricting a token generator to certain virtual server IDs. This optional setting can be applied to each token generator added to the connection using virtual server IDs. By default, no restriction is imposed.
To restrict a token generator to a subset of the available virtual server IDs:

1. Select the Restrict Virtual Server IDs checkbox.
2. In the Allowed Virtual Server IDs area, select one or more virtual server IDs that you want to allow for this token generator.
3. Click Next.

To reach this screen for editing:

1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click Virtual Server IDs on the Summary screen.

**Note:** The Virtual Server IDs screen is only available for connections using at least one virtual server ID, see “Federation Server Identification” on page 37.

Retrieving Additional Attributes

For token generation, you can query local user-data stores to help fulfill the token-generator contract, in conjunction with attribute values supplied by the incoming token.

The values supplied by the token are shown under Attribute Contract on the Attribute Retrieval screen.
Chapter 9
WS-Trust STS Configuration

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click Attribute Retrieval on the Summary screen.
   ▶ If you choose to look up additional information, then you will identify a data store and specify lookup queries next (see the next section “Identifying a Data Store” on page 486).
   ▶ If you use only the attributes available (the default), then you will map values for the attribute contract next (see “Mapping Token Attributes” on page 494).

Tip: To determine whether you need to look up additional values, compare the attribute contract against the token-generator contract on the previous screen (see “Selecting a Token Generator Instance” on page 483). If the token-generator contract requires more information, determine whether your local data stores can supply it. (You can also choose to use text constants or expressions for certain information—see “Mapping Token Attributes” on page 494.)

Identifying a Data Store

This portion of the connection configuration allows you to set up search parameters for a data store.
To reach this screen for editing:

1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click Data Store under the Token Generator Mapping & User Lookup tab.
   If this step is not presented, this Token Generator Instance is not configured to look up user attributes in a data store (see “Retrieving Additional Attributes” on page 485).

To define an attribute source:

- Choose an Active Data Store and click Next.

A data-store configuration must be defined under System Settings for use within a connection. If the data store you want is not shown in the drop-down menu, click Manage Data Stores to add it (see “Managing Data Stores” on page 116).

Configuring Attribute Lookup for WS-Trust STS

See the following sections in this manual, depending on the type of data store:

<table>
<thead>
<tr>
<th>Data Store Type</th>
<th>Related Manual Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>JDBC</td>
<td>• “Selecting a Data Table and Columns” on page 488</td>
</tr>
<tr>
<td></td>
<td>• “Configuring a Database Query” on page 489</td>
</tr>
<tr>
<td>LDAP</td>
<td>• “Configuring LDAP Search Parameters” on page 491</td>
</tr>
<tr>
<td></td>
<td>• “Configuring a Directory Filter” on page 492</td>
</tr>
<tr>
<td>Custom</td>
<td>• “Configuring Custom Filters” on page 494</td>
</tr>
<tr>
<td></td>
<td>• “Selecting Custom Data Fields” on page 494</td>
</tr>
</tbody>
</table>
Selecting a Data Table and Columns

When you choose to use a database source for attributes, you follow this path through the configuration steps.

On this screen you begin to specify exactly where additional data can be found to complete the token-generator contract (see “Retrieving Additional Attributes” on page 485). Only one table may be used as a source of data for a JDBC lookup.

Important: (For MySQL users) To allow for table and column names that may contain spaces, PingFederate inserts double quotes around the names at runtime. To avoid SQL syntax errors resulting from the quotes, add the session variable sql_mode=ANSI_QUOTES to the connection string. For example:

```
```

For more information, see:
- [dev.mysql.com/doc/refman/5.0/en/identifiers.html](dev.mysql.com/doc/refman/5.0/en/identifiers.html)

Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema</td>
<td>Lists the table structure that stores information within a database. Some databases, such as Oracle, require selection of a specific schema for a JDBC query. Other databases, such as MySQL, do not require selection of a schema.</td>
</tr>
<tr>
<td>Table</td>
<td>The name of the table contained in the database. Use the drop-down to change the table.</td>
</tr>
<tr>
<td>Columns to return from SELECT</td>
<td>Displays selected table columns. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.</td>
</tr>
</tbody>
</table>
To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click Database Table and Columns under the Token Generator Mapping & User Lookup tab.

To select a database table and columns for queries:
1. Choose a Schema file (when applicable) from the drop-down list.
2. Choose a Table from the drop-down list.
3. Choose a name under Columns to Return from Select and click Add Attribute.

**Tip:** Click Refresh if you are updating an existing configuration and changes may have been made to the database.

Repeat this step for other columns as needed.

**Note:** You do not need to add a column here for it to be used as part of a search filter (see “Configuring an STS Database Filter” next). Add only attributes from which you need actual values to pass in a token.

**Tip:** To determine what attributes to look up during a query, click the View Attribute Contract link to see what information must be collected (see “Specifying an Attribute Contract” on page 481). Then determine what information is coming in from the token processor (see “Retrieving Additional Attributes” on page 485). Information not contained in the token-processor contract may be pulled from the data store look-up query.

### Configuring a Database Query

The JDBC WHERE clause in PingFederate queries the data table you selected to retrieve a record associated with a particular value (or values) from the incoming security token. The clause is in the form:

```
WHERE column1=value1 [AND column2=value2] [OR ...]
```

The left side of the first variable pair uses a column name in the database table you selected (see “Selecting a Data Table and Columns” on page 488).

The right side generally uses values passed in from the incoming SAML token (variables, including the correct formatting, are listed under Assertion Values).
You can also apply additional search criteria from your own database, using any other columns from the targeted table.

Tip: Click “View List of Columns . . .” to see a list from which to copy and paste.

For more information about WHERE clauses, consult your DBMS documentation.

EXAMPLE:

```
userid=${username}
```

In this example `userid` is the name of a column in the JDBC data store. On the right side, `${username}` returns the value of the `username` variable from the IdP token processor.

Important: You must use the `{}` syntax to retrieve the value of the enclosed variable and use single quotation marks around the `{}` characters.

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where</td>
<td>WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: <code>WHERE email='clive@company.com'</code>.</td>
</tr>
</tbody>
</table>

**To reach this screen for editing:**

1. Click the connection name on the Main Menu.
   
   Click Manage All IdP, if needed, to see a full list of connections.

2. Click **WS-Trust STS** under the IdP Connection tab.

3. Click Configure WS-Trust STS.
4. Click **Token Generation** under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).

5. Click **Configure Token Generation**

6. Click **Token Generator Mapping & User Lookup** on the Summary screen.

7. Click the Token Generator Instance Name.

8. Click **Database Filter** from the steps list under the Token Generator Mapping & User Lookup tab.

**To construct the WHERE clause:**

1. Enter the statement in the space provided, following the guidelines and example above.
   The initial WHERE is optional.

2. Ensure the syntax and variable names are correct.
   When you click **Next**, you will map attribute values returned from the database into the security token (see “Mapping Token Attributes” on page 494).

**Configuring LDAP Search Parameters**

When you choose to use an LDAP source for attributes, you follow this path through the configuration steps.

On this screen you specify the branch of your LDAP hierarchy where you want PingFederate to look up user data.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base DN</td>
<td>The base distinguished name of the tree structure in which the search begins. This field is optional if records are located at the LDAP root.</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Determines the node depth of the query. Select Subtree, One level or Object.</td>
</tr>
</tbody>
</table>
To reach this screen for editing:

1. Click the connection name on the Main Menu.
   - Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   - If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click Token Generator Mapping & User Lookup on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click LDAP Directory Search under the Token Generator Mapping & User Lookup tab.

To select LDAP attributes:

1. (Optional) Enter a Base DN.
2. Select a Search Scope.
3. Select a Root Object Class.
4. Under Attributes to return from search, choose an attribute and click Add Attribute.
   - Note that the attribute Subject DN is always returned by default.
5. Repeat the last step for other attributes as needed.
6. (Optional) Change the Search Scope or the Root Object Class if you want attributes from other locations.

**Note:** You do not need to add an attribute here for it to be used in a search filter (see “Configuring a Directory Filter”). Add only attributes from which you need actual values to pass into the outgoing security token.

### Configuring a Directory Filter

The LDAP filter queries the data you selected to retrieve a record associated with a particular value (or values) from the incoming token. The filter is in the form:

```
attribute=${value}
```

The left-side variable is an attribute you selected earlier (see “Configuring LDAP Search Parameters” on page 491).

The right side generally uses values passed in from the incoming SAML token (variables, including the correct syntax, are listed under Assertion Values).
You can also apply additional search criteria from your data store, using any other attributes from the targeted object classes.

**Tip:** Click “View List of Available LDAP Attributes” for a list from which you can copy and paste.

For general information about search filters, consult your LDAP documentation.

---

### Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>Narrows a search to locate requested data by either including or excluding specific records. An LDAP filter includes the attributes in the search and the value or range of values that the search is attempting to match. Searches are conducted by using three components: 1) at least one attribute (attribute data type) to search on, 2) a search filter operator that will determine what to match, and 3) the value of the attribute being sought. Searches must have at least one of each of these three components.</td>
</tr>
</tbody>
</table>

---

**To reach this screen for editing:**

1. Click the connection name on the Main Menu.
   - Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the IdP Connection tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Generation** under the WS-Trust STS tab.
   - If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click **Configure Token Generation**
6. Click **Token Generator Mapping & User Lookup** on the Summary screen.
7. Click the Token Generator Instance Name.
8. Click **LDAP Filter** under the Token Generator Mapping & User Lookup tab.
To construct the LDAP filter:

1. Enter the statement in the space provided, following the guidelines and example above.

   **Note:** If you used an anonymous binding to create this LDAP connection, your access might be restricted (see “Configuring an LDAP Connection” on page 121).

2. Ensure the syntax and variable names are correct.
3. Click **Next**.

Configuring Custom Filters

When you choose to use a custom source for attributes, you follow this path through the configuration steps.

On this screen you specify a filter, or lookup query, for your custom data source. This screen display and the syntax of the filter depends on your developer's implementation of the custom source SDK.

Selecting Custom Data Fields

On the **Configure Custom Source Fields** screen, you can choose from among the fields shown to map to the token processor contract. These choices are supplied by the driver implementation. Select only those needed to fulfill the attribute contract for this partner connection.

Mapping Token Attributes

You map attributes for outgoing security tokens for this partner on the **Token Generator Contract Fulfillment** screen.

<table>
<thead>
<tr>
<th>Token Generator Instance</th>
<th>Attribute Retrieval</th>
<th><strong>Token Generator Contract Fulfillment</strong></th>
<th>Issuance Criteria</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Map each attribute to fulfill the Token Generator Contract from one of these Sources:

- Assertion
  
  When you make this selection, the associated Value drop-down list is populated by the incoming SAML token (“Assertion”).
• Context
Values are returned from the context of the transaction at runtime.

Note: The HTTP Request and STS SSL Client Certificate Chain selections are retrieved as Java objects rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values.

Choose Expression and then click Edit to enter an expression (see “Using the OGNL Edit Screen” on page 577). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 573. For syntax and examples, see sections under “Constructing Expressions” on page 574.)

Note: When using the STS Basic Authentication Username, STS SSL Client Certificate's Subject DN, or STS SSL Client Certificate Chain attributes, ensure the associated authentication is enabled and configured in WS-Trust STS Settings (see “Selecting Authentication Methods” on page 439).

• LDAP/JDBC/Custom
Values are returned from the selected data store (see “Retrieving Additional Attributes” on page 485). When you make this selection, the Value list is populated by the LDAP, JDBC or Custom attributes specified in previous screens (see “Configuring LDAP Search Parameters” on page 491, “Selecting a Data Table and Columns” on page 488, or “Configuring Custom Filters” on page 494).

• Expression (when enabled)
This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see “Using Attribute Mapping Expressions” on page 573). All of the variables available for text entries (see below) are also available for expressions.

• Text
The value is what you enter. This can be text only, or you can mix text with references to any of the values from the incoming token, using the ${{attribute}} syntax.

You can also enter values from your data store, when applicable, using this syntax:

${ds.attribute}

where attribute is any of the data store attributes you select.

To reach this screen for editing:
1. Click the connection name on the Main Menu.
   Click Manage All IdP, if needed, to see a full list of connections.
2. Click WS-Trust STS under the IdP Connection tab.
3. Click Configure WS-Trust STS.
4. Click Token Generation under the WS-Trust STS tab.
   If this step is not shown, token generation is not selected for the connection (see “Configuring STS Protocol Settings” on page 480).
5. Click Configure Token Generation
6. Click **Token Generator Mapping & User Lookup** on the Summary screen.

7. Click the Token Generator Instance Name.

8. Click **Token Generator Contract Fulfillment** under the Token Generator Mapping & User Lookup tab.

**To map attributes:**

1. Choose a Source for each Target attribute.

2. Choose (or enter) a Value for each Attribute.

   See “Map each attribute to fulfill the Token Generator Contract from one of these Sources:” above. All values must be mapped.

3. Click **Next**.

**Selecting Issuance Criteria for Token Generation (Optional)**

Use this screen to define criteria PingFederate can evaluate to determine whether to issue a security token for a user (see “About Token Authorization” on page 24). This token authorization can be used to restrict who can access identity-enabled Web services.

**To configure Issuance Criteria:**

1. Select a source containing attributes for token authorization.

   Associated attributes appear in the Attribute Name drop-down list:
   - **Assertion** – Select to access attributes from the assertion.
   - **Context** – Select to use values returned from the context of the transaction at runtime.
   - **LDAP/JDBC/Custom** (if a data store is configured) – Select to access attributes returned from a data source.

   **Note:** The HTTP Request and STS SSL Client Certificate Chain selections are retrieved as Java objects rather than text. For this reason, OGNL expressions are more appropriate to evaluate and return values. If you want to use an expression for authentication, skip to Step 8 below.

   **Note:** PingFederate appends a description in parentheses for **Mapped Attributes** (see “Configuring STS Attribute Sources and User Lookup” on page 460).

2. **LDAP/JDBC/Custom** (if a data store is configured) – Select to access attributes returned from a data source.

   **Note:** PingFederate appends a description in parentheses for **Mapped Attributes** (see “Configuring STS Attribute Sources and User Lookup” on page 460).
2. Select an attribute name.

   **Note:** When using the STS Basic Authentication Username, STS SSL Client Certificate's Subject DN, or STS SSL Client Certificate Chain attributes, ensure the associated authentication is enabled and configured in WS-Trust STS Settings (see “Selecting Authentication Methods” on page 439).

3. Select the Condition you want to apply.

   **Note:** Use multi-value conditions when you want PingFederate to verify that an attribute contains or does not contain the specified value in its attribute list. Using a single-value condition when an attribute has multiple values causes the condition to evaluate consistently to false.

   **Important:** When using the STS SSL Client Certificate’s Subject DN attribute, you must select one of the following conditions: equal to DN, not equal to DN, multi-value contains DN, or multi-value does not contain DN. These operators normalize the DN before comparison to accommodate for different string representations that are still considered equivalent (for example, case sensitivity, or whitespace).

4. Enter a value for the attribute.

5. (Optional) Use the Error Result field to customize an error code or error message for use if the token authorization fails.

   Errors result in a SOAP message returned. The Error Result field is used by the faultstring element for SOAP 1.1 and the Reason/Text element for SOAP 1.2.


   **Note:** Using an error code in the Error Result field allows an application to process the code in a variety of ways—for example, display an error message or e-mail an administrator.

   If you leave this field blank, a default ACCESS_DENIED error result is used if the authorization fails.

6. Click **Add**.

7. Repeat the steps above to add additional attributes, as needed, for each authorization criteria.

   **Note:** All criteria must pass in order for a user to be authorized.
8. (Optional) Click **Show Advanced Criteria** to configure more complex evaluations using OGNL expressions.

**Note:** Expressions must be enabled for the **Show Advanced Criteria** button to appear (see “Enabling and Disabling Expressions” on page 573).

**Important:** When you multiplex one connection for multiple environments (see “Connecting to a Partner in One Connection” on page 38), consider using an OGNL expression to verify the virtual server ID in conjunction with other conditions, such as group membership information, to protect against unauthorized access (see “Expression Examples” on page 576).

- Use the in-line editor box to enter the OGNL expression.
  
  For more information about OGNL, see “Using Attribute Mapping Expressions” on page 573.
- Use the **Error Result** box to enter an error code or an error message for use if authorization fails (see step 5 above for more information).

**Note:** If the OGNL expression resolves to a string value (rather than true or false), then the returned value overrides the Error Result.

- Click **Add**. Continue to add expressions, as needed.
- Click the **Test** link to input values and test the resulting output for the expression.

**Note:** For more information on testing OGNL expressions, see “Using the OGNL Edit Screen” on page 577. For syntax and examples, see sections under “Constructing Expressions”.

9. Click **Next** (or **Done**, if you are finished with the configuration, and then **Save** on the Manage Mappings screen).

**To edit Issuance Criteria:**

- Click **Edit** under Actions for the criteria.

**To delete Issuance Criteria:**

- Click **Delete** under Actions for the criteria and then click **Save**.

**Saving the Mapping**

When you have finished configuring Token Generator Mapping & User Lookup, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.

- If you are editing an existing connection, click **Done** on successive screens until you reach the WS-Trust STS screen, and then click **Save**.

**To save a new configuration:**

1. Click **Done** to return to the Token Generator Mapping & User Lookup screen.
2. Click **Next** to go to the Token Generation Summary screen, and then click **Done**.
3. On the Token Generation screen, click **Done**.
4. On the WS-Trust STS screen, click **Save**.

**Using the Token Generation Summary Screen**

When you have finished configuring Token Generation, you can review the configuration on the Summary screen. If you need to make any changes, click the heading over the information you want to edit.

- If you are editing an existing connection, click **Done** on successive screens until you reach the WS-Trust STS screen, and then click **Save**.

On the WS-Trust STS Summary screen, you can review the configuration for this connection.

- If you need to make any changes to a new or existing connection, click the heading over the information you want to modify.
OpenToken Adapter Configuration

In order to transfer identity and other user information between the PingFederate server and an end application, the product architecture allows for custom adapters to be deployed with the server (see “SSO Integration Kits and Adapters” on page 14).

PingFederate ships with a deployed OpenToken Adapter, which uses a secure token format (OpenToken) to transfer user attributes between an application and the PingFederate server. On the IdP side, the OpenToken Adapter allows the PingFederate server to receive a user's identity from the IdP application. On the SP side, the OpenToken Adapter can be used to transfer user-identity information to the target SP application.

Specialized application integration kits are available from www.pingidentity.com. Many kits leverage the OpenToken Adapter to integrate applications with the PingFederate server. The agent portions of the integration kits reside with the application and use the OpenToken to communicate with the OpenToken Adapter.

**Note:** To integrate applications for use with the OpenToken Adapter, download an integration kit for PingFederate from www.pingidentity.com and follow instructions for installing and using Agent Toolkits in the accompanying documentation. Follow the configuration instructions in this appendix to set up the OpenToken Adapter to use with your applications.
Appendix A
OpenToken Adapter Configuration

The following figure shows a basic IdP-initiated SSO scenario using PingFederate with the Java Integration Kit on both sides of an identity federation.

![Diagram of SSO scenario]

**Figure 4:** SP-Initiated SSO: POST/POST

**Processing Steps:**
1. A user initiates an SSO transaction.
2. The IdP application inserts attributes into the Agent Toolkit for Java, which encrypts the data internally and generates an OpenToken.
3. A request containing the OpenToken is redirected to the PingFederate IdP server.
4. The server invokes the OpenToken IdP Adapter, which retrieves the OpenToken, decrypts, parses, and passes it to the PingFederate IdP server. The PingFederate IdP server then generates a Security Assertion Markup Language (SAML) assertion.
5. The SAML assertion is sent to the SP site.
6. The PingFederate SP server parses the SAML assertion and passes the user attributes to the OpenToken SP Adapter. The Adapter encrypts the data internally and generates an OpenToken.
7. A request containing the OpenToken is redirected to the SP application.
8. The Agent Toolkit for Java decrypts and parses the OpenToken and makes the attributes available to the SP Application.

**Configuring the IdP OpenToken Adapter**

1. If you have not already done so, log on to the PingFederate administrative console and click **Adapters** under IdP Configuration on the Main Menu.
2. On the Manage IdP Adapter Instances screen, click **Create New Instance**.
3. On the adapter Type screen, enter an Instance Name and Instance Id, select OpenToken Adapter 2.5.1 (or higher) as the Type, and click Next.

   The Instance Id may not contain spaces or underscores.

4. (Optional) Select a Parent Instance from the drop-down list.

   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

5. Click Next.

6. On the IdP Adapter screen, enter the values as described for the adapter configuration.

   These values are dependent on your developer's implementation.

7. (Optional) Click Show Advanced Fields to reconfigure default settings for the OpenToken, as needed.

   Refer to the on-screen field descriptions and following table for more information.

---

**Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.
Appendix A
OpenToken Adapter Configuration

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Mode</td>
<td>How the token is transported to/from the application, either via a query parameter, a cookie (default), or as a form POST. NOTE: Form POST is applicable only for an SP adapter instance.</td>
</tr>
<tr>
<td>Token Name</td>
<td>The name of the cookie or query parameter that contains the token. This name must be unique for each adapter instance.</td>
</tr>
<tr>
<td>Cipher Suite</td>
<td>The algorithm, cipher mode, and key size that should be used for encrypting the token.</td>
</tr>
<tr>
<td>Logout Service</td>
<td>The URL to which the user is redirected for a single-logout event. This URL is part of an external application, which terminates the user session.</td>
</tr>
<tr>
<td>Cookie Domain</td>
<td>The server domain, preceded by a period (for example, .example.com). If no domain is specified, the value is obtained from the request.</td>
</tr>
<tr>
<td>Cookie Path</td>
<td>The path for the cookie that contains the token.</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>The duration (in seconds) for which the token is valid. Valid range is 1 to 28800.</td>
</tr>
<tr>
<td>Session Lifetime</td>
<td>The duration (in seconds) for which the token may be re-issued without authentication. Valid range is 1 to 259200.</td>
</tr>
<tr>
<td>Not Before Tolerance</td>
<td>The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.</td>
</tr>
<tr>
<td>Force SunJCE Provider</td>
<td>If selected, the SunJCE provider is forced for encryption/decryption.</td>
</tr>
<tr>
<td>Use Verbose Error Messages</td>
<td>If selected, use verbose TokenException messages.</td>
</tr>
<tr>
<td>Obfuscate Password</td>
<td>If selected, the password is obfuscated and password-strength validation is applied. Clearing the checkbox allows backward compatibility with previous OpenToken agents.</td>
</tr>
<tr>
<td>Session Cookie</td>
<td>If selected, OpenToken is set as a session cookie (rather than a persistent cookie). Applies only if Transport Mode is set as 'Cookie'.</td>
</tr>
<tr>
<td>Secure Cookie</td>
<td>If selected, the OpenToken cookie is set only if the request is on a secure channel (https). Applies only if Transport Mode is set as 'Cookie'.</td>
</tr>
<tr>
<td>Delete Cookie</td>
<td>If selected, the token cookie is deleted immediately after consumption. This option is valid only if Transport Mode is set to 'Cookie'.</td>
</tr>
</tbody>
</table>
8. Click Next.


10. On the next screen, click Export and save the properties file.

   The values in the resulting file, agent-config.txt, represent the console configuration and are used by the IdP application. Refer to your respective Integration Kit User Guide for more information.

11. Click Next.

12. (Optional) On the Extended Contract screen, you can configure additional attributes for the adapter (see “Extending an Adapter Contract” on page 233).

   Note: If this is a child instance, select the override checkbox to modify the configuration.

13. Click Next.
Appendix A
Open Token Adapter Configuration

14. On the Adapter Attributes screen, select the subject checkbox under Pseudonym (optionally, select other attributes, if you added any at Step 12). This selection is used if any of your SP partners make use of pseudonyms for account linking (see “Account Linking” on page 17).

**Note:** A selection is required regardless of whether you use pseudonyms for account linking. This allows account linking to be used later without having to delete and reconfigure the adapter. Ensure that you choose at least one attribute that is unique for each user (for example, email) to prevent the same pseudonym from being assigned to multiple users.

You can also choose to mask the values of any or all attributes that PingFederate logs from the adapter at runtime (see “Attribute Masking” on page 23). If OGNL expressions might be used to map derived values into outgoing assertions and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

15. Click Next.

16. On the Summary screen, review the configuration and click Done. You can also click any heading to go back and change information.

17. On the Manage IdP Adapter Instances screen, click Save.

**Important:** You must click Save to retain the adapter configuration.

---

### Configuring the SP OpenToken Adapter

1. If you have not already done so, log on to the PingFederate administrative console and click Adapters under SP Configuration on the Main Menu.

2. On the Manage Adapter Instances screen, click Create New Adapter Instance.
3. On the adapter Type screen, enter an Instance Name and Instance Id, select OpenToken Adapter 2.5.1 (or higher) as the Type, and click Next.
   The Instance Id may not contain spaces or underscores.

4. (Optional) Select a Parent Instance from the drop-down list.
   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

5. Click Next.

6. Enter values for the adapter configuration on the Instance Configuration screen.
   These values are dependent on your developer's implementation.

7. (Optional) Click Show Advanced Fields to reconfigure default settings for the OpenToken, as needed.
   Refer to the on-screen descriptions and the following table for more information.

---

**Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.
## OpenToken Adapter Configuration

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Mode</td>
<td>How the token is transported to/from the application, either via a query parameter, a cookie, or as a form POST (default).</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Form POST is applicable only for an SP adapter instance.</td>
</tr>
<tr>
<td>Token Name</td>
<td>The name of the cookie or query parameter that contains the token. This name must be unique for each adapter instance.</td>
</tr>
<tr>
<td>Cipher Suite</td>
<td>The algorithm, cipher mode, and key size that should be used for encrypting the token.</td>
</tr>
<tr>
<td>Authentication Service</td>
<td>The URL to which the user is redirected for an SSO event. This URL overrides the Target Resource which is sent as a parameter to the Authentication Service.</td>
</tr>
<tr>
<td>Account Link Service</td>
<td>The URL to which the user is redirected for Account Linking. This URL is part of an external SP application. This external application performs user authentication and returns the local user ID inside the token.</td>
</tr>
<tr>
<td>Logout Service</td>
<td>The URL to which the user is redirected for a single-logout event. This URL is part of an external application, which terminates the user session.</td>
</tr>
<tr>
<td>Cookie Domain</td>
<td>The server domain, preceded by a period (for example, .example.com). If no domain is specified, the value is obtained from the request.</td>
</tr>
<tr>
<td>Cookie Path</td>
<td>The path for the cookie that contains the token.</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>The duration (in seconds) for which the token is valid. Valid range is 1 to 28800.</td>
</tr>
<tr>
<td>Session Lifetime</td>
<td>The duration (in seconds) for which the token may be re-issued without authentication. Valid range is 1 to 259200.</td>
</tr>
<tr>
<td>Not Before Tolerance</td>
<td>The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.</td>
</tr>
<tr>
<td>Force SunJCE Provider</td>
<td>If selected, the SunJCE provider is forced for encryption/decryption.</td>
</tr>
<tr>
<td>Obfuscate Password</td>
<td>If selected the password is obfuscated and password-strength validation is applied. Clearing the checkbox allows backward compatibility with previous OpenToken agents.</td>
</tr>
<tr>
<td>Session Cookie</td>
<td>If selected, OpenToken is set as a session cookie (rather than a persistent cookie). Applies only if Transport Mode is set as ‘Cookie’.</td>
</tr>
</tbody>
</table>
8. Click **Next**.

9. On the Actions screen, click **Download** under Action Invocation Link.

10. On the next screen, click **Export** and save the properties file.
    
    The values in the resulting file, `agent-config.txt`, are set by the console configuration and used by the SP application. Refer to your respective Integration Kit User Guide for more information.

11. Click **Next**.

12. (Optional) On the Extended Contract screen, you can configure additional attributes for the adapter (see “Extending Adapter Contracts” on page 341).

    **Note:** If this is a child instance, select the override checkbox to modify the configuration.

13. Click **Next**.

14. On the Summary screen, review the configuration and click **Done**.
    
    You can also click any heading to go back and change information.

15. On the Manage Adapter Instances screen, click **Save**.

    **Important:** You must click **Save** to retain the adapter configuration.
Note: If this is the second instance of an OpenToken Adapter configuration, then you must first click **Next** and map target URLs to adapter instances (see “Mapping URLs to Adapter Instances” on page 342).
HTTP Basic Adapter Configuration

Initial user authentication is normally handled outside of the PingFederate server using an application or IdM system logon module. PingFederate's adapter and application agents are typically used to integrate with these local authentication mechanisms (see “SSO Integration Kits and Adapters” on page 14).

PingFederate packages an HTTP Basic Adapter that delegates user authentication to a configured password credential validator (see “Validating Password Credentials” on page 217). This authentication mechanism validates credentials based on either an LDAP directory or a simple username validator that authenticates credentials maintained by PingFederate.

On the IdP side, when the PingFederate IdP server receives an authentication request for SP-initiated SSO or the user clicks a link for IdP-initiated SSO, the IdP server invokes the HTTP Basic Adapter and, if not already authenticated, prompts the user for local IdP credentials. The credentials are then validated using the designated password credential validator and, if validated, a SAML assertion is generated.

Configuring the HTTP Basic IdP Adapter

1. If you have not already done so, configure a password credential validator (see “Validating Password Credentials” on page 217).
2. Click Adapters under IdP Configuration on the Main Menu screen.
3. On the Manage IdP Adapter Instances screen, click Create New Instance.
4. On the adapter Type screen, enter an Instance Name and Instance Id, select HTTP Basic IdP Adapter as the Type.
Appendix B
HTTP Basic Adapter Configuration

5. **(Optional) Select a Parent Instance** from the drop-down list.

   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see "Hierarchical Plug-in Configurations" on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

6. **Click Next.**
7. On the IdP Adapter screen, click Add a new row to ‘Credential Validators’ to define a credential-authentication mechanism instance for the adapter.

8. Select a password credential validator from the list and click Update. Add as many validators as necessary. Use Move Up and Move Down to adjust the order in which you want PingFederate to attempt credential authentication. If the first mechanism fails to validate the credentials, PingFederate moves sequentially through the list until credential validation succeeds. If none of the defined password credential validators is able to authenticate the user's credentials, and the challenge retries maximum has been reached, the process fails.

9. Enter values for the adapter configuration, as described below.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Realm           | (Required) The name of a protected area. The value of this field is sent as a part of the HTTP Basic authentication request. It appears in a dialog box that prompts the user for a username and password.  
**Note:** Once a user authenticates against a realm, if additional HTTP Basic adapters have the same realm, the user is not prompted to re-authenticate. |

| Challenge Retries | (Required) The number of attempts allowed for password authentication. |
10. Click **Next**.

*Note:* If this is a child instance, select the override checkbox to modify the configuration.

11. On the Adapter Attributes screen, select the username checkbox under Pseudonym (and, optionally, other attributes, if available).

This selection is used if any of your SP partners use pseudonyms for account linking (see “Account Linking” on page 17).

*Note:* A selection is required regardless of whether you use pseudonyms for account linking. This allows account linking to be used later without having to delete and reconfigure the adapter. Ensure that you choose at least one attribute that is unique for each user (for example, email) to prevent the same pseudonym from being assigned to multiple users.

You can also choose to mask the values of any or all attributes that PingFederate logs from the adapter at runtime (see “Attribute Masking” on page 23).

If OGNL expressions might be used to map derived values into outgoing assertions and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

12. Click **Next**.
13. On the Summary screen, review the configuration and click **Done**.
   You can also click any heading to go back and change information.

14. On the Manage IdP Adapter Instances screen, click **Save**.

**Important:** You must click **Save** if you want to retain the adapter configuration.
Appendix B
HTTP Basic Adapter Configuration
Initial user authentication is normally handled outside of the PingFederate server using an application or IdM system logon module. PingFederate's adapter and application agents are typically used to integrate with these local authentication mechanisms (see “SSO Integration Kits and Adapters” on page 14).

PingFederate packages an HTML Form Adapter that delegates user authentication to a configured password credential validator (see “Validating Password Credentials” on page 217). This authentication mechanism validates credentials based on either an LDAP directory or a simple username validator that authenticates credentials maintained by PingFederate.

On the IdP side, when the PingFederate IdP server receives an authentication request for SP-initiated SSO or the user clicks a link for IdP-initiated SSO, the IdP server invokes the HTML Form Adapter and, if not already authenticated, prompts the user for local IdP credentials. The credentials are then validated using the designated password credential validator and, if validated, a SAML assertion is generated.

The HTML Form Adapter allows you to customize a different login page for each configured adapter instance. You can define a logout path and page or a logout redirect page. You can also enable users to change their network passwords and customize a change-password page, or redirect users to a company-hosted password management system.

Configuring the HTML Form IdP Adapter

1. If you have not already done so, configure a password credential validator (see “Validating Password Credentials” on page 217).
2. Click Adapters under IdP Configuration on the Main Menu screen.
3. On the Manage IdP Adapter Instances screen, click Create New Instance.
4. On the adapter Type screen, enter an Instance Name and Instance Id, select HTML Form IdP Adapter as the Type.
Appendix C

HTML Form Adapter Configuration

5. (Optional) Select a Parent Instance from the drop-down list.

If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

6. Click Next.
7. On the IdP Adapter screen, click **Add a new row to ‘Credential Validators’** to define a credential-authentication mechanism instance for the adapter.

8. Select a password credential validator from the list and click **Update**.

Add as many validators as necessary. Use Move Up and Move Down to adjust the order in which you want PingFederate to attempt credential authentication. If the first mechanism fails to validate the credentials, PingFederate moves sequentially through the list until credential validation succeeds. If none of the defined password credential validators is able to authenticate the user’s credentials, and the challenge retries maximum has been reached, the process fails.

---

**Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.
9. Enter values for the adapter configuration, as described below.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Challenge Retries</td>
<td>(Required) The number of attempts allowed for password authentication.</td>
</tr>
<tr>
<td>Session State</td>
<td>How the session state is maintained between adapters of the same type. When Globally is selected, sessions are shared among other HTML Form IdP Adapter instances that use the same ‘Globally’ setting. When Per Adapter is selected, a session is tied to a specific adapter instance. When None is selected, a session is not maintained.</td>
</tr>
<tr>
<td>Session Timeout</td>
<td>Number of idle minutes before the session times out. Leave the field blank to bind the lifetime of the session to that of the PF cookie. <strong>Note:</strong> By default, the PF cookie is a session cookie; web browsers typically remove session cookies on exit. For more information, see “Extending the Lifetime of the PF Cookie” on page 94.</td>
</tr>
<tr>
<td>Login Template</td>
<td>(Required) Template on the IdP server that displays when prompting the user for their credentials. PingFederate allows each configured adapter instance to use a different login page template.</td>
</tr>
<tr>
<td>Logout Path</td>
<td>Any path in the format indicated. Setting a path invokes adapter logout functionality that is normally invoked during SAML 2.0 single-logout processing. Available primarily for partner SaaS providers who do not support SAML Single Log-out (SLO) but who may want users’ IdP SSO sessions to end after logging out of the SaaS services.</td>
</tr>
<tr>
<td>Logout Redirect</td>
<td>The landing page at the SP after successful IdP logout (applicable only when Logout Path is set above).</td>
</tr>
<tr>
<td>Logout Template</td>
<td>Template on the IdP server to display after successful IdP logout, if Logout Redirect fails or is not provided (applicable only when Logout Path is set above).</td>
</tr>
<tr>
<td>Allow Password Changes</td>
<td>Enables or disables the ability for users to change their network passwords using this adapter instance. Select the checkbox to enable the change password functionality. <strong>Note:</strong> The LDAP Password Credential Validator (PCV) is currently the only PCV packaged with PingFederate that supports the change password feature. You can build custom PCVs using the PingFederate Software Development Kit (see the SDK Developer’s Guide for more information).</td>
</tr>
<tr>
<td>Change Password Template</td>
<td>Template on the IdP server that displays when prompting users to change their password. PingFederate allows each configured adapter instance to use a different change password template.</td>
</tr>
</tbody>
</table>
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change Password Message Template</td>
<td>Template on the IdP server that notifies users their password was successfully changed.</td>
</tr>
<tr>
<td>Password Management System URL</td>
<td>URL for redirecting users to a company-specific password management system to change their password.</td>
</tr>
<tr>
<td>Password Management System Message Template</td>
<td>Template on the IdP server that notifies users they are being redirected to a password management system to change their password.</td>
</tr>
<tr>
<td>Login Challenge Template</td>
<td>Displays a configurable challenge form for two-step authentication. This template can be used, for example, to create a RADIUS challenge form when using the RADIUS Username/Password Credential Validator.</td>
</tr>
<tr>
<td>Enable 'Remember My Username'</td>
<td>Allows users to store their username as a cookie when authenticating with this adapter. Once stored, the username is pre-populated in the login form’s username field on subsequent transactions. Select the checkbox to enable the cookie functionality.</td>
</tr>
<tr>
<td>'Remember My Username’ Lifetime</td>
<td>Number of days the cookie remains valid. Enter the number of days you want the username remembered in a cookie. The default is 30. The cookie lifetime is reset upon each successful login in which the Enable ‘Remember My Username’ checkbox is selected.</td>
</tr>
</tbody>
</table>

10. Click **Next**.
11. (Optional) On the Extended Contract screen, you can configure additional attributes for the adapter (see “Extending an Adapter Contract” on page 233).

**Tip:** Any attributes available from credential validators may be specified here for use during SSO authentication processing without explicitly being defined within the PCV instance itself. For example, if you are using the RADIUS credential validator and need additional user attributes returned from its successful authentication, enter the attributes here. Be sure to use the exact attribute names returned by the credential validator.

**Note:** If this is a child instance, select the override checkbox to modify the configuration.

12. Click **Next**.

13. On the Adapter Attributes screen, select the username checkbox under Pseudonym (and, optionally, other attributes, if available).

This selection is used if any of your SP partners use pseudonyms for account linking (see “Account Linking” on page 17).

**Note:** A selection is required regardless of whether you use pseudonyms for account linking. This allows account linking to be used later without having to delete and reconfigure the adapter. Ensure that you choose at least one attribute that is unique for each user (for example, email) to prevent the same pseudonym from being assigned to multiple users.

You can also choose to mask the values of any or all attributes that PingFederate logs from the adapter at runtime (see “Attribute Masking” on page 23).

If OGNL expressions might be used to map derived values into outgoing assertions and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

14. Click **Next**.

15. On the Summary screen, review the configuration and click **Done**.

You can also click any heading to go back and change information.
16. On the Manage IdP Adapter Instances screen, click **Save**.

---

**Important**: You must click **Save** if you want to retain the adapter configuration.
Composite Adapter Configuration

For an IdP, PingFederate includes a Composite Adapter, which allows an administrator to “chain” the selection of available adapter instances for a connection. At runtime, adapter chaining means that SSO requests are passed sequentially through each adapter instance specified until one or more authentication results are found for the user.

Adapter chaining may be used to choose an adapter instance based on the method by which a user authenticated, or to integrate an organization's multi-factor authentication policy.

Configuring the Composite Adapter

1. If you have not already done so, configure instances of IdP Adapters you want to use for adapter chaining (see “Configuring IdP Adapters” on page 230).
2. Click Adapters under IdP Configuration on the Main Menu screen.
3. On the Manage Adapter Instances screen, click Create New Adapter Instance.
Appendix D

Composite Adapter Configuration

4. On the adapter Type screen, enter an Instance Name and Instance Id, select Composite Adapter as the Type.

   The Instance Id may not contain spaces or underscores.

5. (Optional) Select a Parent Instance from the drop-down list.

   If you are creating an instance that is similar to an existing instance, you might consider making it a child instance by specifying a parent (see “Hierarchical Plug-in Configurations” on page 17). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

6. Click Next.

<table>
<thead>
<tr>
<th>Type</th>
<th>Manage IdP Adapter Instances</th>
<th>Create Adapter Instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td>Manage IdP Adapter Instances</td>
<td>Create Adapter Instance</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ADAPTER</th>
<th>POLICY</th>
<th>AUTHN CONTEXT WEIGHT</th>
<th>AUTHN CONTEXT OVERRIDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML Form</td>
<td>Required</td>
<td>Sufficient</td>
<td>3</td>
</tr>
<tr>
<td>Reference IdP Adapter</td>
<td>Required</td>
<td>Sufficient</td>
<td>3</td>
</tr>
</tbody>
</table>

**Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.

7. On the IdP Adapter screen, click **Add a new row to ‘Adapters’**.

   **Note:** This screen may appear differently than the example above, depending on types of adapters configured on your system (see Step 14).

8. Select an Adapter Instance from the drop-down list.

   If a required adapter instance is not shown in the list, save or cancel this configuration and ensure that the instance has been configured and saved (see “Configuring IdP Adapters” on page 230).

9. (Optional) Change or enter different defaults under any or all of the adjacent column headings under Adapters, as described in the following table.

   **Note:** If this is a child instance, select the override checkbox related to the settings you want to modify.

### Attribute Name Synonyms

<table>
<thead>
<tr>
<th>NAME</th>
<th>SYNONYM</th>
</tr>
</thead>
</table>

**Field Name | Field Value | Description**

<table>
<thead>
<tr>
<th>ATTRIBUTE INSERTION</th>
<th>Add To Back</th>
<th>Add To Front</th>
</tr>
</thead>
</table>

**Definitions:**
- Add To Back: Adds the value to the back of the list.
- Add To Front: Adds the value to the front of the list.
- Defines the order in which different values are returned for the same attribute name.
10. Add at least one more Adapter Instance.

11. As needed, use the Move Down/Move Up links under Action to re-order the entries.

    At runtime adapter chaining is sequential, starting at the top of the list.

    **Important:** Several types of adapters—for example, the Integrated Windows Authentication Adapter—may be configured to direct end users to an error page if authentication fails for any reason, which will halt further progress through a composite-adapter chain. Ensure that for such adapter instances the “Error URL” option is *not* used in the instance configuration, if continuation through an adapter-chaining sequence is required.

12. (Optional) If any attributes are logically equivalent across two adapter instances but have different names, click Add a new row to ‘Attribute Name Synonyms’ and enter the attribute names under Name and Synonym.

    The attribute name under Synonym and its value are used in the SAML assertion, when the two values returned from each adapter are identical. If
13. (Optional) Change the Field Value selected for Attribute Insertion.

For attributes of the same name configured in different adapter instances, you can change the order of returned values when the values are different. (Values are merged if they are the same.)

By default (Add to Back) the value for an attribute name configured in the first instance is returned first and also listed first in the resulting SAML assertion. Then any different value from the same attribute name in a subsequently invoked instance is appended.

The order might not matter for many attributes, but in the case of the SAML-subject attribute, only the first value in the SAML assertion may be used for an SP connection partner under normal circumstances. Click Add to Front to reverse the default order, if needed.

14. If any IdP Adapter developed with version 2 (or later) of the Adapter SDK is installed and configured on your system, complete the entries under Input User Id Mapping.

This section appears only for such adapters (for example, the adapter included with the separately available Verisign Identity Protection Integration Kit). For two-factor authentication these adapters may require that a unique ID be passed in from a first-factor adapter. If so, an administrator must specify the attribute containing the unique ID on this screen. (For basic information about the Adapter SDK, see the SDK Developer's Guide. For specific developer information about the version 2 IdP-adapter interface, refer to the Javadoc located in the <pf_install>/pingfederate/sdk/doc directory.)

15. Click Next.

16. On the Extended Contract screen, Add attributes to be returned from each adapter instance configured on the previous screen.

   Important: Attributes must correspond exactly to any or all of the attribute names listed on the Adapter Attribute screens for each configured adapter instance.

17. Click Next.
18. On the Adapter Attributes screen, select at least one attribute as a Pseudonym (and, optionally, other attributes, if available).

This selection is used if any of your SP partners make use of pseudonyms for account linking (see “Account Linking” on page 17).

**Note:** A selection is required regardless of whether you use pseudonyms for account linking. This allows account linking to be used later without having to delete and reconfigure the adapter. Ensure that you choose at least one attribute that is unique for each user (for example, email) to prevent the same pseudonym from being assigned to multiple users.

You can also choose to mask the values of any or all attributes that PingFederate logs from the adapter at runtime (see “Attribute Masking” on page 23).

If OGNL expressions might be used to map derived values into outgoing assertions and you want those values masked, select the related checkbox under the Attribute list (see “Using Attribute Mapping Expressions” on page 573).

19. Click **Next**.

20. On the Summary screen, review the configuration and click **Done**.

You can also click any heading to go back and change information.

21. On the Manage IdP Adapter Instances screen, click **Save**.

**Important:** You must click **Save** if you want to retain the adapter configuration.
Application Endpoints

These endpoints provide a means, via standard HTTP, by which external applications can communicate with the PingFederate server.

The SSO and SLO endpoints for an IdP and an SP include optional parameters which you can use to specify error pages that users see in the event of an SSO or SLO failure. By default, PingFederate provides templates for these and other errors or conditions (see “Customizing User-Facing Screens” on page 87).

SP endpoints also include those available for SCIM Inbound Provisioning (see “Provisioning for SPs” on page 35).

For either SP or IdP servers, a maintenance endpoint is also provided for administrators to verify that the server is running. Endpoints applicable to both server roles also include those needed for adapter-to-adapter mapping (see “IdP-to-SP Adapter Mapping” on page 136) and retrieval of WS-Trust metadata (see “WSC and WSP Support” on page 8).

IdP Endpoints

The following sections describe PingFederate IdP endpoints, including the query parameters that each accepts or requires. These endpoints accept either the HTTP GET or POST methods.

**Note:** Begin each URL with the fully qualified server name and port number of your IdP or SP PingFederate server: for example: 

Important: When the parameter TargetResource (or TARGET) is used and includes its own query parameters, the parameter value must be URL-encoded. Any other parameters that contain restricted characters (many SAML URNs, for example) also must be URL-encoded.

For information about URL encoding, see, for example, “HTML URL-encoding Reference” (www.w3schools.com/tags/ref_urlencode.asp).

Note: Parameters are case-sensitive.

/idp/startSSO.ping

This is the path used to initiate an unsolicited IdP-initiated SSO transaction during which a SAML response containing an assertion is sent to an SP. Typically, a systems integrator or developer creates one or more links to this endpoint in the IdP application or portal to allow users to initiate SSO to various SPs.

For information about allowing applications to retrieve configuration data from the PingFederate server over SOAP, see “Web Service Interfaces” on page 553.

The following table shows the HTTP parameters for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PartnerSpId</td>
<td>The federation ID of the SP to whom the SAML response containing an assertion should be issued. One of these parameters is required unless the federation ID can be derived from TargetResource or TARGET (see below).</td>
</tr>
<tr>
<td>or PARTNER</td>
<td></td>
</tr>
<tr>
<td>TargetResource</td>
<td>For SAML 2.0, the value of either parameter is passed to the SP as the RelayState element of a SAML response message. This is the PingFederate implementation of the SAML 2.0 indicator for a desired resource at the SP during IdP-initiated SSO. For SAML 1.x, the value is sent to the SP as a parameter named TARGET. Note: If this parameter is not provided in the URL, then the target resource should be specified in the administrative console (see “Configuring a Default URL and Error Message” on page 243).</td>
</tr>
<tr>
<td>or TARGET</td>
<td></td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
</tr>
<tr>
<td>InErrorResource</td>
<td>Indicates where the user is redirected after an unsuccessful SSO. If this parameter is not included in the request, PingFederate redirects the user to the SSO error landing page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87).</td>
</tr>
<tr>
<td>(optional)</td>
<td></td>
</tr>
</tbody>
</table>
/idp/startSLO.ping

This is the path used to initiate an IdP-initiated SLO (under SAML 2.0) or an OpenID Connect logout (see “Asynchronous Front-Channel Logout” on page 157). Typically, a systems integrator or developer creates one or more links to this endpoint in the protected resources of their IdP application or portal to allow users to end their sessions at various SPs. This endpoint uses the local PingFederate session to determine which SPs have been issued an SSO assertion and sends them a SAML logout request.
The following table shows the HTTP parameters for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetResource (optional)</td>
<td>Indicates where the user is redirected after a successful SLO. If this parameter is not included in the request, PingFederate uses as a default the URL for a successful SLO as entered on the IdP Default URL screen.</td>
</tr>
<tr>
<td>InErrorResource (optional)</td>
<td>Indicates where the user is redirected after an unsuccessful SLO. If this parameter is not included in the request, PingFederate redirects the user to the SLO error landing page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87).</td>
</tr>
<tr>
<td>Binding (optional - SAML 2.0)</td>
<td>Indicates the binding to be used; allowed values are URIs defined in the SAML specifications. The SAML 2.0 applicable URIs are: urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact, urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST, urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect, urn:oasis:names:tc:SAML:2.0:bindings:SOAP. When the parameter is not used, the first SLO Service URL configured for the SP-partner connection is used (see “Specifying SLO Service URLs (SAML 2.0)” on page 294).</td>
</tr>
</tbody>
</table>

/idp/writecdc.ping

This endpoint is used for SAML 2.0 IdP Discovery. This is the path used when the IdP wants to write to the Common Domain Cookie (CDC) held within the user’s browser. The information written to the cookie indicates from which IdP this user has authenticated.

The following table shows the one HTTP query parameter for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetResource (optional)</td>
<td>Indicates where the user is redirected after successful IdP Discovery. If this parameter is not included in the request, PingFederate redirects the user to the referrer in the HTTP header. If there is no TargetResource or referrer, the call to this endpoint will fail.</td>
</tr>
</tbody>
</table>
The following sections describe the PingFederate SP endpoints, including the query parameters that each accepts or requires.

**Note:** Begin each URL with the fully qualified server name and port number of your IdP or SP PingFederate server: for example: https://pingidentity.com:9031/idp/startSSO.ping.

### SP Services
These endpoints accept either the HTTP GET or POST methods.

**Important:** When the parameter TargetResource is used and includes its own query parameters, the parameter value must be URL-encoded. For information about URL encoding, see, for example, “HTML URL-Encoding Reference” (http://www.w3schools.com/tags/ref_urlencode.asp).

**Note:** Parameters are case-sensitive.

### /sp/startSSO.ping
This is the path used to initiate SP-initiated SSO. In this scenario, the SP issues an SSO request to the IdP asking for an SSO authentication response. Typically, a systems integrator or developer creates one or more links to this endpoint in SP applications to allow users to access various protected resources via SSO using the IdP as an authentication authority.

For information about allowing applications to retrieve configuration data from the PingFederate server over SOAP, see “Web Service Interfaces” on page 553.

The following table shows the HTTP parameters for this endpoint.

**Note:** Some parameters described below can have multiple values. Specify these values by using multiple independent query string parameters of the same name.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| PartnerIdpId (required if more than one IdP connection is configured and Domain is not used) | The federation ID of the IdP that authenticates the user and issues an assertion.  
**Note:** This ID is case-sensitive. |
| Domain (required to invoke Auto-Connect) | The domain name associated with the requesting user’s IdP (see “Using Auto-Connect” on page 30). In this case, PartnerIdpId cannot be used. |
### Application Endpoints

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| **TargetResource** or **TARGET** (optional) | This parameter indicates where the end-user is redirected after a successful SSO.  
**Note:** When this parameter is not provided in the URL, a default target resource may be specified in the administrative console, either for all IdP connections (see “Configuring Default URLs” on page 346) or for individual connections (see “Configuring Default Target URLs (Optional)” on page 398), or both. |
| **Binding** (optional)       | Indicates the binding to be used; allowed values are URIs defined in the SAML specifications. For example, the SAML 2.0 applicable URIs are:  
urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact  
urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST  
urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect  
When the parameter is not used for SAML 2.0, the first SSO Service URL configured for the IdP-partner connection is used (see “Specifying SSO Service URLs (SAML)” on page 393). |
<p>| <strong>InErrorResource</strong> (optional) | This parameter indicates where the end-user is redirected after an unsuccessful SSO. If this parameter is not included in the request, PingFederate redirects the user to the SLO error landing page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87). |
| <strong>SpSessionAuthn AdapterId</strong> (optional) | The explicit SP adapter instance ID indicating the adapter to use to create an authenticated session or security context. |
| <strong>ForceAuthn</strong> (optional - SAML 2.0) | This parameter controls the attribute of the same name in the AuthnRequest. (The default is false.) |
| <strong>IsPassive</strong> (optional - SAML 2.0) | This parameter controls the attribute of the same name in the AuthnRequest. (The default is false.) |
| <strong>AllowCreate</strong> (optional - SAML 2.0) | Controls the value of the AllowCreate attribute of the NameIDPolicy element in the AuthnRequest. (The default is true.) |
| <strong>RequestedFormat</strong> (optional - SAML 2.0) | Specifies the value for the Format attribute in the NameIDPolicy element of the AuthnRequest. If not specified, the attribute is not included in the AuthnRequest. |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthenticatingIdpId (optional - SAML 2.0)</td>
<td>This parameter indicates the preferred IdP for authenticating the user through an IdP proxy. The parameter specifies the value of the IDP Entry ProviderID attribute in the Scoping element of the AuthnRequest. Multiple values are permitted in order to build a preferred list.</td>
</tr>
<tr>
<td>RequestedACSIIdx (optional - SAML 2.0)</td>
<td>The index number of your site’s Assertion Consumer Service, where you want the assertion to be sent.</td>
</tr>
<tr>
<td>RequestedAcsUrl (optional - SAML 2.0)</td>
<td>The URL of your site’s Assertion Consumer Service, where you want the assertion to be sent.</td>
</tr>
<tr>
<td>RequestedBinding (optional - SAML 2.0)</td>
<td>Indicates the binding requested for the response containing the assertion; allowed values are URIs defined in the SAML specifications.</td>
</tr>
<tr>
<td>RequestedAuthnCtx (optional - SAML 2.0)</td>
<td>Indicates the requested authentication context of the assertion; allowed values include URIs defined in the SAML specifications (see the OASIS SAML document saml-authn-context-2.0-os.pdf). Multiple values are permitted in order to build a preferred list.</td>
</tr>
<tr>
<td>RequestedAuthnDeclRef (optional - SAML 2.0)</td>
<td>An alternative to RequestedAuthnCtx, above, indicating the requested authentication context of the assertion by declaring any URI reference (see section 2.7.2.2 of the OASIS SAML document saml-core-2.0-os.pdf). Multiple values are permitted in order to build a preferred list.</td>
</tr>
<tr>
<td>RequestedSPName Qualifier (optional - SAML 2.0)</td>
<td>Indicates that the IdP should return the given name qualifier as part of the assertion (used primarily to identify SP affiliations—see “Defining SP Affiliations” on page 329).</td>
</tr>
<tr>
<td>vsid</td>
<td>Specify the virtual server ID. When absent, PingFederate uses the default virtual server ID (if specified) for the connection (see “General Connection Information” on page 360) or the SAML federation ID defined in Server Settings (see “Specifying Federation Information” on page 108).</td>
</tr>
</tbody>
</table>

If an adapter is specified in SpSessionAuthnAdapterId, then that adapter is used to create an authenticated session for SP-initiated SSO. If there is no SpSessionAuthnAdapterId, the ultimate destination of the user after SSO (either the TargetResource or the default SSO success URL) is used along with
the mappings defined in the administrative console on the Map URLs to Adapter Instances screen (see “Mapping URLs to Adapter Instances” on page 342).

Note that adapter selection for SP-initiated SSO is similar to that for IdP-initiated SSO except that, because the adapter ID is dependent on the SAML deployment, PingFederate cannot expect it from an IdP. Therefore, it uses only the URL mapping for adapter selection for SSO.

/sp/startSLO.ping

This is the path used to initiate SP-initiated SLO. Typically, a systems integrator or developer creates one or more links to this endpoint in the protected resources of their SP application, which allows users to end a session by sending a logout request to the IdP that authenticated the session.

Note that the IdP might send additional logout request messages to other SPs when it receives a logout request from a PingFederate server acting as an SP.

The following table shows the HTTP parameters for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetResource (optional)</td>
<td>Indicates where the user is redirected after a successful SLO. If this parameter is not included in the request, PingFederate uses as a default the URL for a successful SLO, as entered on the SP Default URLs screen.</td>
</tr>
<tr>
<td>Binding (optional - SAML 2.0)</td>
<td>Indicates the binding to be used; allowed values are URIs defined in the SAML specifications. The SAML 2.0 applicable URIs are:</td>
</tr>
<tr>
<td></td>
<td>urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact</td>
</tr>
<tr>
<td></td>
<td>urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST</td>
</tr>
<tr>
<td></td>
<td>urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect</td>
</tr>
<tr>
<td></td>
<td>urn:oasis:names:tc:SAML:2.0:bindings:SOAP</td>
</tr>
<tr>
<td></td>
<td>When the parameter is not used, the first SLO Service URL configured for the IdP-partner connection is used (see “Specifying SLO Service URLs” on page 395).</td>
</tr>
<tr>
<td>InErrorResource (optional)</td>
<td>Indicates where the user is redirected after an unsuccessful SLO. If this parameter is not included in the request, PingFederate redirects the user to the SLO error landing page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87).</td>
</tr>
<tr>
<td>SpSessionAuthn AdapterId (optional)</td>
<td>The SP adapter instance ID indicating which session to terminate and which IdP will receive the logout request.</td>
</tr>
<tr>
<td>SourceResource (optional)</td>
<td>A URL indicating the origin of the logout request. It is mapped to an adapter ID in order to designate which session to terminate.</td>
</tr>
</tbody>
</table>
An SP PingFederate session can be associated with one or more application sessions relying on any number of IdPs as the session authority. PingFederate must choose one session to terminate and also send an SLO request to the IdP that issued the assertion that created the session. Sessions are associated with the ID of the adapter instance that created them. Once an adapter ID is determined, the first session found with that ID is used. Determination of the adapter instance ID occurs in the following order:

1. If there is a value for the `SpSessionAuthnAdapterId` parameter, it is used.
2. If there is a value for the `SourceResource` parameter, PingFederate attempts to map a URL to an adapter using that value to determine the adapter ID.
3. If there is an HTTP header value for `Referer [sic]`, PingFederate attempts to map a URL to an adapter using that value to determine the adapter ID.
4. If none of the above is successful, the `TargetResource` parameter value or the value for the default SLO success URL are used to map a URL to an adapter.
5. Finally, if no adapter ID is determined, the first one in the list is used.

### /sp/defederate.ping

This is the path used to terminate an account link created during SSO. Account linking provides a means for subject identification on the SP side. Links are created and terminated entirely by a user on the SP side. The link contains the name identifier from the IdP, the IdP’s federation ID, the adapter instance ID, and the local user identifier.

There are no HTTP parameters for this endpoint.

You can unlink a user session only if was established during SSO using an existing account link on the SP side. If more than one SP session was established via account linking on the same PingFederate session, each of those links will be terminated by this endpoint. A local logout is also performed for any link that is terminated.

### /sp/cdcstartSSO.ping

This endpoint is used for IdP-Discovery implementations (see “IdP Discovery” in the “Supported Standards” chapter of Getting Started). This endpoint is similar to `/sp/startSSO.ping` and accepts the same parameters, with the exception of `PartnerIdpId` and `vsid` (see “/sp/startSSO.ping” on page 533). Instead of this parameter, the server attempts to use the common domain cookie to determine the IdP.

### /sp/startAttributeQuery.ping

This endpoint is used to initiate an Attribute Query with a SAML 2.0 IdP (see “Attribute Query and XASP” in the “Supported Standards” chapter of Getting Started).

The following table shows the HTTP parameters for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>Uniquely identifies the user to the IdP. When user authenticates with an x.509 certificate, this is the Subject DN, which must be URL-encoded.</td>
</tr>
</tbody>
</table>
## Application Endpoints

### SCIM Inbound Provisioning Endpoints

The following sections describe the PingFederate endpoints for SCIM Inbound Provisioning (see “Inbound Provisioning” on page 35).

These endpoints are defined in the following SCIM 1.1 specifications:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Issuer (optional)             | The IssuerDN from the user’s x.509 certificate (when XASP is used), which uniquely identifies the entity that issued the user’s certificate. The parameter must be URL-encoded.  
  **Note**: When specified this parameter overrides the Subject parameter. |
| PartnerIdpId (except for XASP) | Used to identify the specific IdP partner to which the Attribute Query should be sent. If this parameter is not present, the Subject and Issuer are used to determine the correct IdP.  
  **Note**: For XASP, this parameter overrides both the Subject and Issuer parameters. |
| Format (required for XASP, otherwise optional) | Identifies the name-identifier format of the Subject query parameter. If included, the value must be one of the SAML 2.0 Name Identifier Format URLs (see section 8.3 of the SAML specifications [http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf](http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf)).  
  **Note**: For XASP, this parameter must be set to: urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName  
  If not specified, the parameter defaults to: urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified.  
  **Note**: The parameter must be URL-encoded. |
| AppId                          | The unique identifier of the initiating application.                                                                                                                                                   |
| SharedSecret                  | Used to authenticate the initiating application. The AppId and SharedSecret must both match the application authentication settings within the PingFederate server.                                                  |
| RequestedAttrName (optional)  | A name of a user attribute requested from the IdP. For each such desired user attribute, include this parameter. If this parameter is not present, then all allowable user attributes are returned from the IdP.  
  Multiple values are permitted in order to build a preferred list. |
| vsid (optional)               | Specify the virtual server ID.  
  When absent, PingFederate uses the default virtual server ID (if specified) for the connection (see “General Information” on page 255) or the SAML federation ID defined in Server Settings (see “Specifying Federation Information” on page 108). |
SP Endpoints

- **SCIM Core Schema** (www.simplecloud.info/specs/draft-scim-core-schema-01.html)
- **SCIM Specification** (www.simplecloud.info/specs/draft-scim-api-01.html)

**Note:** The PingFederate implementation is described in the next section, “/pf-scim/v1/Users”. Developers can find additional information about the implementation via the Service Provider Configuration Endpoint (see “/pf-scim/v1/ServiceProviderConfigs” on page 540).

/pf-scim/v1/Users

The Users endpoint is where client applications make HTTP requests to create, retrieve, update, and delete (or deactivate) users. This REST-based endpoint accepts POST, GET, PUT, and DELETE methods, as described in the following table.

**Note:** HTTP requests must be made using either Basic or client-certificate application authentication (see “Configuring Inbound Provisioning” on page 413). JSON is currently the only supported format for the HTTP message body.

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Description</th>
</tr>
</thead>
</table>
| POST        | Sends user attributes in JSON format—defined in the SCIM Core Schema—to create a new user. If the user is successfully provisioned, the HTTP response indicates a 201 status code and contains a JSON body indicating all of the user attributes added to the datastore. The user ID is set as the id attribute in the JSON response, and the full URL to reference the user is in the HTTP response Location header. For an existing user, you can also use POST either to update or delete/disable user records by appending the user ID to the path (see the Note below) and setting the request header X-HTTP-Method-Override value to PUT or DELETE, respectively. (For more information, see the PUT and DELETE method descriptions below.)

**Note:** The following methods require a user ID parameter appended to the endpoint path: /pf-scim/v1/Users/user_id

| GET         | Retrieves all attributes for the specified user. A successful response is indicated by an HTTP 200 status code as well as the attributes. |
Appendix E
Application Endpoints

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT</td>
<td>Updates user attributes for the specified user, using JSON in the body of the HTTP request. Attributes not included in the request are set to a default value in the data store. A successful PUT operation returns an HTTP 200 status code and the entire updated user record within the response body.</td>
</tr>
<tr>
<td>DELETE</td>
<td>Deletes or disables the user record for the specified user. Whether a user is deleted or disabled is determined by the connection configuration (see “Handling SCIM Delete Requests” on page 423). A successful response is indicated by an HTTP 200 status code.</td>
</tr>
</tbody>
</table>

**Note:** For a list of HTTP error codes that may be returned, see the SCIM specifications (www.simplecloud.info/specs/draft-scim-api-01.html#anchor6).

/pf-scim/v1/ServiceProviderConfigs

This Service Provider Configuration Endpoint is where a client can retrieve detailed information on the PingFederate SCIM 1.1 implementation. When Inbound Provisioning is enabled for an SP PingFederate server, an HTTP GET request to this endpoint returns a JSON response outlining SCIM 1.1 compliance details.

**System-Services Endpoints**

These endpoints apply to the PingFederate server generally, whether used as an IdP, SP, or both.

**Note:** Parameters are case-sensitive.

/pf/heartbeat.ping

This endpoint returns an “OK” browser message and an HTTP 200 status indication if the PingFederate server is running. If you receive an HTTP 404 error, the server associated with the endpoint is down.

Load balancers can use this endpoint to determine the status of PingFederate independently of checks used to determine the status of the supporting hardware.

You can also configure the server to provide regular status information to a network-management utility (see “Configuring Runtime Reporting” on page 100).

/pf/adapter2adapter.ping

This endpoint initiates direct IdP-to-SP adapter mapping, when that feature is configured (see “IdP-to-SP Adapter Mapping” on page 136).
The following table shows the HTTP parameters for this endpoint.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetResource</td>
<td>Indicates where the user is redirected after a successful SSO. If this parameter is not included in the request, PingFederate uses as a default the URL for a successful SSO (see “Configuring Default URLs” on page 346).</td>
</tr>
<tr>
<td>SpSessionAuthn AdapterId</td>
<td>The SP adapter instance ID to be used. If not provided and more than one SP adapter instance is configured with adapter-to-adapter mapping, PingFederate uses configured defaults (see “Mapping URLs to Adapter Instances” on page 342).</td>
</tr>
<tr>
<td>IdpAdapterId</td>
<td>Indicates the IdP adapter to use for authentication if more than one IdP adapter is configured in adapter-to-adapter mappings.</td>
</tr>
<tr>
<td>InErrorResource</td>
<td>Indicates where the user is redirected if the SSO is unsuccessful. If this parameter is not included in the request, PingFederate redirects the user to the SSO error landing page hosted within PingFederate (see “Customizing User-Facing Screens” on page 87).</td>
</tr>
</tbody>
</table>

/pf/sts_mex.ping

This endpoint returns STS metadata for use in expediting configuration of Web-service applications.

The following table shows the HTTP parameters for this endpoint:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PartnerSpId</td>
<td>The Connection ID of the SP to whom the SAML token will be issued. This parameter determines the connection for which metadata will be generated.</td>
</tr>
<tr>
<td>PartnerIdpId</td>
<td>The Connection ID of the IdP issuing the SAML token to be consumed by PingFederate. This parameter determines the connection for which the metadata will be generated.</td>
</tr>
<tr>
<td>vsid</td>
<td>Specify the virtual server ID. If absent, PingFederate uses the default virtual server ID (if specified) for the connection or the federation ID defined in Server Settings (see “Specifying Federation Information” on page 108).</td>
</tr>
</tbody>
</table>

/pf/federation_metadata.ping

This endpoint returns WS-Federation metadata.
The following table shows the HTTP parameters for this endpoint:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PartnerSpId</td>
<td>The Connection ID of the SP to whom the SAML token is issued. This parameter determines the connection for which metadata is generated.</td>
</tr>
<tr>
<td>PartnerIdpId</td>
<td>The Connection ID of the IdP issuing the SAML token to be consumed by PingFederate. This parameter determines the connection for which the metadata is generated.</td>
</tr>
<tr>
<td>vsid</td>
<td>Specify the virtual server ID. If absent, PingFederate uses the default virtual server ID (if specified) for the connection or the federation ID defined in Server Settings (see “Specifying Federation Information” on page 108).</td>
</tr>
</tbody>
</table>

Note: If your partner fails to retrieve metadata when sending both the PartnerSpId (or the PartnerIdpId) and the vsid query parameters, perhaps it is only capable of sending one query parameter in such requests. An alternative metadata exchange endpoint that includes the virtual server ID information should resolve the issue.

**To construct a metadata exchange endpoint for a specific virtual server ID:**

1. Construct a JSON object containing a key-value pair of the virtual server ID as follows:
   
   ```json
   {"vsid":"<VirtualServerIdValue>"
   
   Example: {"vsid":"Engineering"}
   
   2. Base64url-encode the JSON object. 
   
   Example: eyJ2c2lkIjoiRW5naW51ZXJpbmcifQ
   
   3. Insert the base64url-encoded value between /pf/ and /federation_metadata.ping (or /sts_mex.ping).
   
   Example: /pf/eyJ2c2lkIjoiRW5naW51ZXJpbmcifQ/federation_metadata.ping (or /pf/eyJ2c2lkIjoiRW5naW51ZXJpbmcifQ/sts_mex.ping)
   
   For more information about base64url, see RFC4648 (tools.ietf.org/html/rfc4648).
OAuth 2.0 Endpoints

The following sections describe OAuth-developer information on PingFederate endpoints for the OAuth AS.

**Note:** Unless otherwise indicated, these endpoints and associated parameters are defined in the OAuth 2.0 Authorization Protocol (see “OAuth 2.0” in the “Supported Standards” chapter of Getting Started).

**Note:** Begin each URL with the fully qualified server name and port number of your IdP or SP PingFederate server: for example: `https://pingidentity.com:9031/as/token.oauth2`.

### Token Endpoint

The token endpoint is defined in the OAuth 2.0 specification and used by the client to obtain an access token and possibly a refresh token by presenting its authorization grant. The token endpoint is used with every authorization grant except for the Implicit grant type (since an access token is issued directly from the authorization endpoint).

**Endpoint:** `/as/token.oauth2`

**Note:** By default per OAuth specifications, this endpoint accepts only the HTTP POST method.

### Client Identification and Authentication

Clients can authenticate to the OAuth AS using this endpoint by presenting their client identifier and client secret either using the HTTP Basic authentication...
OAuth 2.0 Endpoints

scheme (where the client identifier is the username, and the client secret is the password) or with the following HTTP request parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client_id</td>
<td>The client identifier (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>client_secret</td>
<td>The client secret (as defined on the client management UI screen).</td>
</tr>
</tbody>
</table>

Whenever possible, the use of HTTP Basic is recommended over the use of the request parameters.

Clients without a client secret can use the client_id parameter to identify themselves to the OAuth AS and omit the client_secret parameter.

Grant Type Parameters

Other parameters accepted by the /as/token.oauth2 endpoint vary by the grant type being presented and include both OAuth-defined standard parameters and PingFederate-specific parameters. The grant type of the access token request is indicated by the following parameter:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grant_type</td>
<td>Indicates the type of grant being presented in exchange for an access token and possibly a refresh token. The value is an extensibility mechanism of the OAuth 2.0 specification. PingFederate supports these values:</td>
</tr>
<tr>
<td></td>
<td>• authorization_code</td>
</tr>
<tr>
<td></td>
<td>• password</td>
</tr>
<tr>
<td></td>
<td>• client_credentials</td>
</tr>
<tr>
<td></td>
<td>• refresh_token</td>
</tr>
<tr>
<td></td>
<td>• urn:ietf:params:oauth:grant-type:saml2-bearer</td>
</tr>
<tr>
<td></td>
<td>• urn:pingidentity.com:oauth2:grant_type:validate_bearer</td>
</tr>
</tbody>
</table>

Note: Further parameters associated with each grant type are defined in the following sections.

Authorization Code Grant Type

These parameters apply when the grant_type parameter for /as/token.oauth2 is set to authorization_code.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>The authorization code received from the authorization server during the redirect interaction at the authorization endpoint when the response_type parameter is code (see “Endpoint: /as/authorization.oauth2” on page 549).</td>
</tr>
</tbody>
</table>
### Token Endpoint

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refresh_token</td>
<td>The refresh token issued to the client during a previous access-token request.</td>
</tr>
<tr>
<td>scope</td>
<td>The scope of the access request expressed as a list of space-delimited, case-sensitive strings. The requested scope must be equal to or less than the scope originally granted by the resource owner. If omitted, the scope is treated as equal to that originally granted by the resource owner. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 159). Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 177).</td>
</tr>
</tbody>
</table>

### Refresh Token Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `refresh_token`.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refresh_token</td>
<td>The refresh token issued to the client during a previous access-token request.</td>
</tr>
</tbody>
</table>

### Resource Owner Credentials (Password) Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `password`.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>The username, encoded as UTF-8.</td>
</tr>
<tr>
<td>password</td>
<td>The password, encoded as UTF-8.</td>
</tr>
<tr>
<td>scope</td>
<td>The scope of the access request. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 159). Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 177).</td>
</tr>
</tbody>
</table>
### OAuth 2.0 Endpoints

#### Client Credentials Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `client_credentials`.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validator_id</td>
<td>A PingFederate OAuth AS parameter indicating the instance ID of the password credential validator to be used to check the username and password (and the associated attribute mapping into the <code>USER_KEY</code> of the persistent grant). If multiple validator instances are configured and mapped and no <code>validator_id</code> is provided, each instance will be tried sequentially until one succeeds or they all fail.</td>
</tr>
</tbody>
</table>

Client authentication is required, which means either HTTP Basic or `client_id` and `client_secret` must be included (see “Client Identification and Authentication” on page 543).

#### SAML 2.0 Bearer Assertion Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `urn:ietf:params:oauth:grant-type:saml2-bearer`.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assertion</td>
<td>A single SAML 2.0 assertion, which must be encoded using base64url, as described in Section 5 of RFC4648 (<a href="http://tools.ietf.org/html/rfc4648#section-5">http://tools.ietf.org/html/rfc4648#section-5</a>).</td>
</tr>
<tr>
<td>scope</td>
<td>The scope of the access request. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 159). Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 177).</td>
</tr>
</tbody>
</table>
Access Token Verification/Validation Grant Type

These parameters apply when the `grant_type` parameter for

`/as/token.oauth2` is set to

`urn:pingidentity.com:oauth2:grant_type:validate_bearer`.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token</td>
<td>The bearer access token to be validated.</td>
</tr>
</tbody>
</table>

The Validation grant type is a custom PingFederate OAuth extension that enables an RS to communicate with the OAuth AS while leveraging the established communication and encoding patterns from OAuth 2.0. The grant type allows an RS to check with the OAuth AS on the validity of a bearer access token that it has received from a client making a protected-resources call.

Client authentication is not required. For this grant type, the RS acts in the role of a client for the request/response exchange with the OAuth AS to make the validation call.

The response is a standard OAuth access-token response from the token endpoint with some extensions and minor semantic differences in the treatment of some of the parameters. The returned token is in a JSON structure with name-to-value attributes or name-to-array attributes.

The token type is `urn:pingidentity.com:oauth2:validated_token`—a URN indicating the token represents the attributes associated with the validated access token passed on the request. A `client_id` parameter is returned indicating the client identifier of the client to whom the grant was made. A `scope` parameter is returned, if the scope is greater than the default implied scope, indicating the approved scope of the grant. The `expires_in` parameter indicates for how many more seconds the token is valid (note that the value may increase on subsequent validation calls if a token lifetime extension policy is in place: see “Configuring Reference-Token Management” on page 165).

For example:

```
{
    "scope":"read edit admin",
    "token_type":"urn:pingidentity.com:oauth2:validated_token",
    "expires_in":3172,
    "client_id":"super_cool_mobile_client",
    "access_token":
    {
        "uid":"sfHqhad9onMjXsQNI1mZP9mD7AQasmskd",
        "group":["employee","sales","manager"],
        "email":"someguy@example.cloud"
    }
}
```

Token Revocation Endpoint

The token revocation endpoint is defined in the OAuth 2.0 Token Revocation (RFC 7009) specification. It allows clients to notify the authorization server that a previously obtained refresh or access token is no longer needed. The revocation request invalidates the actual token and possibly other tokens based on the same authorization grant.
Endpoint: `/as/revoke_token.oauth2`

**Note:** By default per OAuth specifications, this endpoint accepts only the HTTP POST method.

**Important:** Direct access token revocation is only supported for *Internally Managed Reference Tokens*. Access tokens of the type *JSON Web Token (JWT)* do not support direct revocation. JWT access tokens can only be indirectly revoked if the associated refresh token is revoked, and the JWT’s configuration field Access Grant GUID Claim Name under Access Token Management is set.

### Client Identification and Authentication for Token Revocation

Clients can authenticate to the OAuth AS using this endpoint by presenting their client identifier and client secret either using the HTTP Basic authentication scheme (where the client identifier is the username, and the client secret is the password) or with the following HTTP request parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client_id</td>
<td>(optional) The client identifier (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>client_secret</td>
<td>(optional) The client secret (as defined on the client management UI screen).</td>
</tr>
</tbody>
</table>

Whenever possible, the use of HTTP Basic is recommended over the use of the request parameters.

Clients without a client secret can use the `client_id` parameter to identify themselves to the OAuth AS and omit the `client_secret` parameter.

The only time a `client_id` is not required is when Unidentified Clients are allowed in Authorization Server Settings. In this case, only tokens issued by unidentified clients will be revoked.

### Parameters

The token revocation endpoint uses the following parameters using the "application/x-www-form-urlencoded" format in the HTTP request entity-body:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token (required)</td>
<td>The token that the client wants to revoke.</td>
</tr>
<tr>
<td>token_type_hint</td>
<td>A hint about the type of token submitted for revocation. PingFederate supports the following values</td>
</tr>
<tr>
<td></td>
<td>• access_token</td>
</tr>
<tr>
<td></td>
<td>• refresh_token</td>
</tr>
</tbody>
</table>
If a refresh token is revoked, then the associated access grant and access tokens will be revoked as well. If an access token is revoked, the associated access grant and refresh token remain untouched with the exception of Implicit grant types. If Reuse Existing Persistent Access Grants for GrantTypes is enabled in Authorization Server Settings then the Implicit access grant will also be revoked with the access token.

Authorization Endpoint

The authorization endpoint is defined in the OAuth 2.0 specification and is used by the OAuth AS to interact directly with resource owners, authenticate them, and obtain their authorization. Typically, an OAuth client makes an authorization request by directing a resource owner, via an HTTP user-agent, to the authorization endpoint. After completing its interaction with the resource owner, the OAuth AS redirects the resource owner's user-agent back to the client's redirect URI with the response to the authorization request.

Note: This endpoint may be used as part of an OAuth Scope Selector configuration (see “Configuring the OAuth Scope Selector” on page 239), which can affect the behavior of the endpoint. For example, the IdP parameter might be enforced or overridden by policy determined by the OAuth Scope Selector.

Endpoint: /as/authorization.oauth2

The table below shows parameters for this endpoint:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response_type</td>
<td>A value of code results in the Authorization Code grant type while a value of token implies the Implicit grant type.</td>
</tr>
<tr>
<td>client_id</td>
<td>The client identifier (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>redirect_uri</td>
<td>Required if more than one URI is configured in PingFederate for the client or if a wildcard is used for a single URI entry (see “Configuring a Client” on page 178). Optional for clients with only one specific redirect URI configured.</td>
</tr>
<tr>
<td></td>
<td>Note that if this parameter is used, the same parameter and value must also be used in subsequent token requests (see “Authorization Code Grant Type” on page 544).</td>
</tr>
<tr>
<td>scope</td>
<td>The scope of the access request expressed as a list of space-delimited, case-sensitive strings. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 159).</td>
</tr>
<tr>
<td></td>
<td>Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 177).</td>
</tr>
</tbody>
</table>
OpenID Connect Parameters

The table below displays OpenID Connect parameters for this endpoint:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>An opaque value used by the client to maintain state between the request and callback. If included, the AS returns this parameter and the given value when redirecting the user agent back to the client.</td>
</tr>
<tr>
<td>idp (or PartnerIdpId)</td>
<td>A PingFederate OAuth AS parameter indicating the Entity ID/Connection ID of the IdP with whom to initiate Browser SSO for user authentication.</td>
</tr>
<tr>
<td>pfidpadapterid</td>
<td>A PingFederate OAuth AS parameter indicating the IdP Adapter Instance ID of the adapter to use for user authentication. <strong>Note:</strong> This parameter may be overridden by policy based on adapter selector configuration. For example, the OAuth Scope Selector could enforce the use of a given adapter based on client-requested scopes (see “Configuring Adapter Selectors” on page 235).</td>
</tr>
<tr>
<td>nonce (optional)</td>
<td>Specifies a string value used to associate a Client session with an ID Token and to reduce replay attacks. The value is passed through unmodified from an Authorization Request to the ID Token.</td>
</tr>
<tr>
<td>prompt (optional)</td>
<td>Specifies whether the AS prompts the end user for reauthentication and consent. Expressed as a list of space-delimited, case-sensitive ASCII string values. If included, this parameter can be used by the client to verify that the end user is still present for the current session or to bring attention to the request. PingFederate supports the following values: none, login, consent.</td>
</tr>
<tr>
<td>acr_values (optional)</td>
<td>Specifies the Authentication Context Class Reference (acr) values for the AS to use when processing an Authentication Request. Express as a space-separated string, listing the values in order of preference.</td>
</tr>
</tbody>
</table>

If more than one source of authentication is configured in the system and no pfidpadapterid or idp parameter is provided, users are presented with an intermediate page asking them to choose among the available sources of authentication. The authentication results in a set of user attributes that must be mapped into the USER_KEY attribute for persistent grant storage and the USER_NAME attribute that is displayed on the user authorization page.
Grant-Management Endpoint

The grants endpoint (two are provided, one for use with parameters) is where end-users/resource owners go to view (and optionally revoke) the persistent access grants they have made. This endpoint is not part of the OAuth specification, but many OAuth providers offer a similar type of functionally. The grants displayed are those associated with the \texttt{USER\_KEY} of the authenticated user. The same attribute mapping(s) from the authentication source to \texttt{USER\_KEY} used for the authorization endpoint are used here to look up the user's existing grants.

Endpoints: /as/grants.oauth2 and /as/oauth_access_grants.ping

The table below shows parameters for this endpoint.

\begin{table}[h]
\centering
\begin{tabular}{|l|l|}
\hline
Parameter & Description \\
\hline
idp (or PartnerIdpId) & Indicates the Entity ID/Connection ID of the IdP with whom to initiate Browser SSO for user authentication. \\
\text{(optional)} & \\
\hline
pfidpadapterid & Indicates the IdP Adapter Instance ID of the adapter to use for user authentication. \\
\text{(optional)} & \\
\hline
\end{tabular}
\end{table}

Note: Use only the endpoint /as/grants.oauth2 with these optional parameters.

Important: When a parameter is needed for this endpoint, use only one of these options.

If no recent user attributes are found for the session context, the user is redirected to /as/oauth_access_grants.ping to initiate the authentication process, which behaves in exactly the same way as the authorization endpoint (see “Token Endpoint” on page 543).

OpenID Connect Metadata Endpoint

This public endpoint provides metadata needed for an OAuth client to interface with PingFederate using the OpenID Connect protocol (see “OpenID Connect” on page 14).

Endpoint: /.well-known/openid-configuration

No parameters are needed for this endpoint.
Web Service Interfaces

PingFederate provides two built-in, SOAP-accessible Web Services related to browser SSO. These services may be used by client applications to manage partner connections and support integration of Web applications, respectively:

- **Connection Management Service** — Enables creation and deletion of single connection configurations in PingFederate. This service may be used to migrate connections from one server environment to another (for example, from testing or staging to production) or to create new connections in a single server programmatically.

- **SSO Directory Service** — Provides Web application developers with information regarding partner connections and adapter instances.

**Tip:** PingFederate provides a command-line utility that can be used to export and modify connections, as well as other administrative-console configurations, and then import them to target environments (see “Automating Configuration Migration” on page 78).

- **OAuth Client Management Service** — Useful for managing OAuth client applications, where needed (see “About OAuth” on page 10 and “Client Management” on page 177).

- **OAuth Access Grant Management Service** — Allows retrieval and revocation of access grants.

- **Session Revocation API** — Allows OpenID Connect Clients to query revocation status of their sessions and add end-user sessions to the revocation list.

**Tip:** Applications accessing the Connection Management Service must first authenticate themselves to the PingFederate server. SSO Directory Service authentication is optional by default, but may be required. For more information, see “Authentication” on page 215.
Connection Management Service

The Connection Management Service supports basic connection management capabilities and is accessible only on a PingFederate server running the administrative console. This feature is useful in a variety of circumstances, but the following primary use cases were considered:

- As a utility to migrate changes to a partner connection through staging environments (for example: development, test, production).
  Changes to URLs and keys may be needed to make the connection appropriate to the next environment.
- As a way for an external application to update or delete connections programmatically, or create new ones using an exported connection XML file as a template.

The WAR file for this service, pf-mgmt-ws.war, is located in the pingfederate/server/default/deploy2 directory.

The SOAP-accessible service endpoint is pf-mgmt-ws/ws/ConnectionMigrationMgr.

The Web Services Description Language (WSDL) document describing this service can be retrieved from:
https://<host_server>:<admin_console_port>/pf-mgmt-ws/ws/ConnectionMigrationMgr?wsdl

Exporting a Connection

You can export a connection either manually, using the administrative console, or programmatically, via a call to the Connection Management Service.

In either case, the exported XML complies with the standard SAML 2.0 metadata format, with extensions to capture PingFederate's proprietary configuration. Most connection configuration information is contained in the XML markup, with the exception of global configuration items such as adapter instances, data stores, and keypairs. Adapter instances and data stores are referenced by ID, and keypairs are referenced by the MD5 fingerprint of their X.509 certificate. Public certificates, such as the partner's signature verification certificate, are included completely (base-64 encoded).

Exporting Manually

For information about using the administrative console to export SP connections at an IdP site, see “Via the Manage Connections Screen” on page 249.

For information about exporting IdP connection at an SP site, see “From the Manage Connections Screen” on page 355.
Using the Connection Service

The Connection Web Service exposes the following method for exporting connections:

```java
public string getConnection(
    String entityId,
    String role,) throws IOException
```

**Code Sample**

```java
Service service = new Service();
Call call = (Call)service.createCall();
call.setUsername("username");
call.setPassword("password");
call.setTargetEndpointAddress("https://localhost:9999/pf-mgmt-ws/ws/ConnectionMigrationMgr");
call.setOperationName("getConnection");
Object result = call.invoke(new Object[] {"entityId", "SP"});
```

Importing Connections

Moving a connection from one PingFederate server to another requires care, as the target server must contain the global configuration items (data stores, keypairs, and adapter instances) that the connection references. Changing the references in the XML file—either manually or programmatically—may be necessary to adjust the connection to the target PingFederate environment.

Once required changes are made to the XML file, developers can use the Connection Management Service to import the connection into a different instance of PingFederate.

**Tip:** Alternatively, you can import XML connection files into PingFederate manually by copying them into the directory:

```
<pf_install>/pingfederate/server/default/
data/connection-deployer
```

PingFederate scans this directory periodically and imports connections automatically.

**Caution:** Manually importing a connection always overwrites an existing connection with the same ID (the Web Service provides a switch to disallow this behavior, if desired—see below).

The Web Service exposes the following method for importing connections:

```java
public void saveConnection(
    String xml,
    boolean allowUpdate) throws IOException
```

The xml parameter is the complete representation of the connection retrieved by your application from an exported connection file (and optionally modified).

If allowUpdate is false, the Web Service can be used only to add a new connection. An error occurs if a connection already exists with the same connection ID and federation protocol in the XML. If allowUpdate is true and the connection already exists, it will be overwritten.
**Sample Code**

Below is example client code using the Apache AXIS libraries that invokes this Web Service to create a new connection:

```java
Service service = new Service();
Call call = (Call) service.createCall();
call.setUsername("username");
call.setPassword("password");
String addr = "https://localhost:9999/pf-mgmt-ws/ws/ConnectionMigrationMgr";
call.setTargetEndpointAddress(addr);
call.setOperationName("saveConnection");
String xml = "<EntityDescriptor entityID="some_entity_id" ...";
 boolean allowUpdate = false;
call.invoke(new Object[]{xml, allowUpdate});
```

**Deleting Connections**

The Web Service exposes the following method for connection deletion:

```java
public void deleteConnection(
    String entityId,
    String role) throws IOException
```

The `entityId` parameter is the Connection ID, which identifies the connection to be deleted. The `role` parameter is the connection role—IDP or SP.

**Code Example**

Below is example client code using the Apache AXIS libraries that invokes this Web Service to delete a connection:

```java
Service service = new Service();
Call call = (Call) service.createCall();
call.setUsername("username");
call.setPassword("password");
call.setTargetEndpointAddress("https://localhost:9999/pf-mgmt-ws/ws/ConnectionMigrationMgr");
call.setOperationName("deleteConnection");
call.invoke(new Object[]{"entityId", "SP"});
```

**Cluster Configuration Replication**

A Web Service endpoint is available to replicate the administrative-console configuration to other nodes in a PingFederate cluster. This allows a client of this Web Service to create or update a new connection (or delete a connection) and then push the new configuration to the other cluster nodes.

The service endpoint is:

```
/pf-mgmt-ws/ws/ConfigReplication
```

The WSDL document describing this service can be retrieved from:

```
https://<host_server>:<admin_console_port>/pf-mgmt-ws/ws/ConfigReplication?wsdl
```
The Web Service exposes the following method:
public void replicateConfiguration();

Example Code

Below is example client code using the Apache AXIS libraries that invokes the configuration replication functionality:
Call call2 = (Call) service.createCall();
call2.setUsername("joe");
call2.setPassword("test");
String addr2 = "https://localhost:9999/pf-mgmt-ws/ws/ConfigReplication";
call2.setTargetEndpointAddress(addr2);
call2.setOperationName("replicateConfiguration");
call2.invoke(new Object[]{});

Validation Disclaimer

The import process is not subject to the same rigorous data validation performed by the administrative user interface. Although some checks are made, it is possible to create invalid connections using the connection-migration process. Therefore, because the XML is complex and validation is limited, attempting to create an XML connection from scratch is not recommended. Rather, the administrative console should be used to create the initial connection. That way, changes necessary to the exported connection's XML representation can be held to a minimum, reducing the risk of compromising data integrity.

SSO Directory Service

PingFederate SSO Directory Service allows applications to retrieve configuration data from a runtime PingFederate server. (A PingFederate server in a cluster configured as an administrative console does not support this Web Service.) This service allows Web applications to avoid storing and maintaining the data locally. These types of data can be retrieved:
- A list of IdP partners
- A list of SP partners
- A list of IdP adapter instances
- A list of SP adapter instances

The SSO Directory Service provides information useful for integrating an application with a PingFederate server. It is a way for the application to find out dynamically which partners can be used for SSO. This means applications need not be modified when new partners are configured in PingFederate.

The WAR file for this module, pf-ws.war, is located in the pingfederate/server/default/deploy directory.

Note: If you do not want to allow use of the service, it should not be deployed: remove the WAR file from the deploy directory.

The service endpoint is pf-ws/services/SSODirectoryService.

The WSDL document describing this service can be retrieved from:
http(s)://<pf_runtime_host>:<runtime_port>/pf-ws/services/SSODirectoryService?wsdl
You can retrieve a list using any of the following methods:

- **getIDPList** – Returns a list of active IdP connections configured for SP-initiated SSO. The list contains each IdP's Connection ID and Connection Name.
- **getSPList** – Returns a list of active SP connections configured for IdP-initiated SSO. The list contains each SP's Connection ID and Connection Name.
- **getAdapterInstanceList** – Returns a list of SP adapter instances containing an ID and name.
- **getIdpAdapterInstanceList** – Returns a list of IdP adapter instances containing an ID and name.

**Note:** For either IdP or SP lists, Connection IDs are returned as values for the XML tag `<entityId>`. Connection Names are returned as values for the XML tag `<company>` (see “SOAP Request and Response Example” on page 559).

- **getAdapterInstanceList** – Returns a list of SP adapter instances containing an ID and name.
- **getIdpAdapterInstanceList** – Returns a list of IdP adapter instances containing an ID and name.

**Note:** These methods do not require input parameters.

The service is also available over HTTP. The query string for retrieving any of the lists is:
```
/pf-ws/services/SSODirectoryService?method=<method_name>
```

### Coding Example

When you integrate a Web application with PingFederate, use the SSO Directory Service to generate a connection or adapter list. The code needed to create any of the lists is similar.

The following Java code example retrieves an IdP list from the Web Service. The program calls the `getIDPList` method in the SSO Directory Service to retrieve an IdP list and print it to the console. This example uses the Apache Axis library and includes optional code for authentication to the PingFederate server (see “Authentication” on page 215). We recommend the use of HTTPS when including credentials.

```java
import org.apache.axis.client.Call;
import org.apache.axis.client.Service;
import java.net.URL;
import javax.xml.namespace.QName;
import com.pingidentity.ws.SSOEntity;

public class SSODirectoryClientSample {
    public static void main(String[] args) throws Exception {
        Service service = new Service();
        Call call = (Call) service.createCall();
        call.setUsername("username");
        call.setPassword("pass");
        URL serviceUrl = new URL("https://localhost:9031/pf-ws/services/
                                SSODirectoryService");
        QName qn = new QName("urn:BeanService", "SSOEntity");
```
call.registerTypeMapping(SSOEntity.class, qn,
    new org.apache.axis.encoding.ser.BeanSerializerFactory(    
        SSOEntity.class, qn),

    new org.apache.axis.encoding.ser.BeanDeserializerFactory(        
        SSOEntity.class, qn));

call.setTargetEndpointAddress( serviceUrl );
call.setOperationName( new QName(  
        "http://www.pingidentity.com/servicesSSODirectoryService" ,  
        "getIDPList"));
Object result = call.invoke( new Object[] {} );
if (result instanceof SSOEntity[])  
    {  
        SSOEntity[] idpArray = (SSOEntity[])result;
        for (SSOEntity idp : idpArray)  
        {  
            System.out.println(idp.getEntityId() + " " +  
                idp.getCompany());
        }
    }  
else  
    {  
        System.out.println("Received problem response from server: " + result);
    }
}

SOAP Request and Response Example

A client application must send a SOAP request to the PingFederate server specifying the requested Web Service and the specific method. For example, the following is a typical SOAP request for an IdP list using the SSO Directory Service.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
        <ns1:getIDPList
            soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            xmlns:ns1="https://localhost:9031/ssodir/services/SSODirectoryService"/>
    </soapenv:Body>
</soapenv:Envelope>
```

The PingFederate server's Web Service will return a response containing the list you requested. The following is an example of a typical SOAP response for an IdP list:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
        <ns1:getIDPList
            soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            xmlns:ns1="https://localhost:9031/ssodir/services/SSODirectoryService"/>
    </soapenv:Body>
</soapenv:Envelope>
```
OAuth Client Management Service

PingFederate includes a REST-based Web Service for OAuth client management. The service is provided primarily for organizations with many OAuth clients, allowing programmatic management of OAuth clients, as an alternative to using the administrative console (see “Client Management” on page 177).

Note: This Web Service is active only if client records are managed in a database, rather than in XML files (the installed default storage method). For information on setting up database management for OAuth clients, see “Defining an OAuth Client Data Store” on page 131.
The REST resources in the following sections are URL path extensions of the PingFederate runtime endpoint:

```
http(s)://<pf_runtime_host>:<runtime_port>/pf-ws/rest
```

For example:

```
https://my_corp.com:9031/pf-ws/rest/oauth/clients
```

**Note:** POST and PUT methods described in this section require parameter name/value pairs formatted in JavaScript Object Notation (JSON).

**Important:** Applications must authenticate to the Web Service using HTTP Basic credentials specified in a PingFederate Password Credential Validator (see “Validating Password Credentials” on page 217). The configured Credential Validator, in turn, must be selected in the OAuth AS configuration (see “Authorization Server Settings” on page 159).

**Endpoint: /oauth/clients**

This resource accepts the methods POST, PUT, and GET.

**POST**

**Description**

Creates a new client based on the parameters provided in the request. Parameters correspond to administrative-console fields; for additional information, see “Client Management” on page 177.

The required MIME type is `application/json`.

**JSON Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientId</td>
<td>(Required) A unique ID for the client.</td>
</tr>
<tr>
<td>name</td>
<td>(Required) A descriptive name for the client.</td>
</tr>
<tr>
<td>refreshRolling</td>
<td>Indicates whether a new refresh token is issued with each new access token (see “Field Descriptions” on page 178). Allowed values: true or false. When not provided, the global setting for the AS is used (see “Authorization Server Settings” on page 159).</td>
</tr>
<tr>
<td>redirectUris</td>
<td>The URI(s) to which the OAuth AS redirects the resource owner’s user agent after authorization is obtained. <strong>Required</strong> for Implicit and Authorization Code grant types (see grantTypes below).</td>
</tr>
<tr>
<td>logoUrl</td>
<td>A URL for the logo presented to the user on the grant revocation page.</td>
</tr>
</tbody>
</table>
# Parameter Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td>Client password or phrase, required for the Client Credentials grant type unless client-certificate authentication is needed (see next two parameters).</td>
</tr>
<tr>
<td>clientCertIssuerDn</td>
<td>Client TLS certificate issuer, required for the Client Credentials grant type unless a secret is provided.</td>
</tr>
<tr>
<td>clientCertSubjectDn</td>
<td>Client TLS certificate subject, required for the Client Credentials grant type unless a secret is provided.</td>
</tr>
<tr>
<td>description</td>
<td>A description of what the client application does, displayed in browser when the user is prompted for authorization.</td>
</tr>
<tr>
<td>persistentGrantExpirationType</td>
<td>Indicates whether to override the global setting for the AS (see “Authorization Server Settings” on page 159). Allowed values:</td>
</tr>
<tr>
<td>persistentGrantExpirationTime</td>
<td>An integer representing units of time for storage of persistent grants for this client—use with persistentGrantExpirationTimeUnit (see below) and only when persistentGrantExpirationType is OVERRIDE_SERVER_DEFAULT</td>
</tr>
<tr>
<td>persistentGrantExpirationTimeUnit</td>
<td>Units for the expiration time set in the parameter above (if applicable). Allow values: h – hours, d – days</td>
</tr>
<tr>
<td>bypassApprovalPage</td>
<td>If set to true, user consent to resource access is assumed and the approval page is not presented.</td>
</tr>
</tbody>
</table>
Table 24

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>restrictScopes</td>
<td>If set to true, limits client access to a subset of the scopes defined for the AS (see “Authorization Server Settings” on page 159). Scopes are limited to the default scope and any listed for restrictedScopes (see below). If no scopes are listed and this parameter is true, only the default scope is available for the client.</td>
</tr>
<tr>
<td>restrictedScopes</td>
<td>When used with restrictScopes, limits access to the scope(s) provided in the JSON list, in addition to the default scope.</td>
</tr>
<tr>
<td>grantTypes</td>
<td>One or more grant types allowed for the client (see “Grant Types” on page 11). Allowed values:</td>
</tr>
<tr>
<td></td>
<td>• authorization_code</td>
</tr>
<tr>
<td></td>
<td>• password (Resource Owner Password Credentials)</td>
</tr>
<tr>
<td></td>
<td>• refresh_token (Use with authorization_code and/or password grant types.)</td>
</tr>
<tr>
<td></td>
<td>• client_credentials</td>
</tr>
<tr>
<td></td>
<td>• implicit</td>
</tr>
<tr>
<td></td>
<td>• extension (SAML 2.0 Bearer)</td>
</tr>
<tr>
<td></td>
<td>• urn:pingidentity.com:oauth2:grant_type:validate_bearer (Access Token Validation)</td>
</tr>
<tr>
<td>Notes:</td>
<td>• At least one grant type is required.</td>
</tr>
<tr>
<td></td>
<td>• Separate multiple values with commas.</td>
</tr>
</tbody>
</table>

Example JSON

{"client": [
  {
    "clientId": "12345",
    "name": "Client Doe",
    "refreshRolling": "true",
    "redirectUris": [
      "http://www.url.com",
      "http://www.url2.com"
    ],
    "logoUrl": "http://www.url.com/image.gif",
    "clientCertIssuerDn": "CN=CA, dc=pingidentity, dc=com",
    "clientCertSubjectDn": "cn=MyClient, dc=pingidentity, dc=com",
    "description": "Description of the client",
    "persistentGrantExpirationType": "OVERRIDE_SERVER_DEFAULT"
  }
]}
"persistentGrantExpirationTime":"3",
"persistentGrantExpirationTimeUnit":"d",
"bypassApprovalPage":"true",
"restrictScopes":"true",
"restrictedScopes":{
    "scope 1",
    "scope 2"
}
"grantTypes":{
    "password",
    "refresh_token"
}
}
}

Returns

- 200 – Success
- 400 – Failed To Create Client
  The response contains details as to why the client creation failed.
- 401 – Invalid Credentials
  The user does not exist or is not authorized to create a client.
- 500 – Internal Server Error
  An unknown error has occurred.

PUT

Description

Updates client details for a specified client.

Note: You cannot update a client ID—you must delete the client record and create a new one.

JSON Parameters

The same parameters described for POST apply for PUT with one addition:

forceSecretChange

Use this parameter, set to true, in conjunction with the secret parameter to change a client pass phrase.

Note: If the secret parameter is used without forceSecretChange, the secret value is ignored.

Example JSON

{"client":{
    "clientId":"12345",
    "name":"Client Doe",
    "refreshRolling":"true",
    "redirectUris":{
        "http://www.url1.com",
        "http://www.url2.com"
"logoUrl":"http://www.url.com/image.gif",
"forceSecretChange":"true",
"secret":"mysecretphrase",
"description":"Description of the client",
"persistentGrantExpirationType":"OVERRIDE_SERVER_DEFAULT",
"persistentGrantExpirationTime":"3",
"persistentGrantExpirationTimeUnit":"d",
"bypassApprovalPage":"true",
"restrictScopes":"true",
"restrictedScopes":[
  "scope 1",
  "scope 2"
]
"grantTypes":{
  "password",
  "refresh_token"
}
}]

Returns

- 200 – Success
  The body contains a list of updated clients.
- 400 – Failed To Update Client
  The response contains details as to why the client could not be updated.
- 401 – Invalid Credentials
  The user does not exist or is not authorized to update a client.
- 500 – Internal Server Error
  An unknown error has occurred.

GET

Description

Retrieves details for all OAuth clients.

JSON Parameters

None.

Returns

- 200 – Success
  The body contains JSON data for all clients.

  Note: The parameter refreshRolling is not returned if the AS
global setting is set for a client (the default).

- 400 – Failed To Retrieve Clients
  The response contains details as to why clients could not be retrieved.
- 401 – Invalid Credentials
  The user does not exist or is not authorized.
Chapter G
Web Service Interfaces

- 500 – Internal Server Error
  An unknown error has occurred.

Endpoint: /oauth/clients/id
This resource accepts the methods GET and DELETE.

GET
Description
Retrieves details for the specified client ID.

JSON Parameters
None.

Returns
- 200 – Success
  JSON client parameters are included.

  Note: The parameter refreshRolling is not returned if the AS global setting is set for a client (the default).

- 400 – Failed To Retrieve Client
  The response contains details as to why client could not be retrieved.
- 401 – Invalid Credentials
  The user does not exist or is not authorized.
- 500 – Internal Server Error
  An unknown error has occurred.

DELETE
Description
Deletes records for the specified client ID.

Returns
- 200 – Success
- 400 – Failed To Delete Client
  The response contains details as to why client could not be deleted.
- 401 – Invalid Credentials
  The user does not exist or is not authorized.
- 405 – Method Not Allowed
  The client ID was not specified.
- 500 – Internal Server Error
  An unknown error has occurred.

OAuth Access Grant Management Service

PingFederate includes a REST-based Web Service for OAuth access grant management. This service enables retrieval and revocation of access grants for a particular client or user.
The REST resources are URL path extensions of the PingFederate runtime endpoint:
http(s)://<pf_runtime_host>:<runtime_port>/pf-ws/rest

For example:
https://my_corp.com:9031/pf-ws/rest/oauth/clients

**Important:** Applications must authenticate to the Web Service using HTTP Basic credentials specified in a PingFederate Password Credential Validator (see “Validating Password Credentials” on page 217). The configured Credential Validator, in turn, must be selected in the OAuth AS configuration (see “Authorization Server Settings” on page 159).

Endpoints: /oauth/clients/{clientId}/grants[/{grantId}] and /oauth/users/{userKey}/grants[/{grantId}]

These resources accept the methods GET which supports retrieval and DELETE which supports revocation of access grants.

**Note:** The Clients endpoint requires client records to be managed in a database, rather than in XML files (the installed default storage method). For information on setting up database management for OAuth clients, see “Defining an OAuth Client Data Store” on page 131.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clientId</td>
<td>The client identifier to retrieve or revoke grants for (see “Configuring a Client” on page 178).</td>
</tr>
<tr>
<td>userKey</td>
<td>The user key to retrieve or revoke grants for. The user key is defined under IdP Adapter Mapping and Resource Owner Credentials Mapping.</td>
</tr>
<tr>
<td>grantId (optional)</td>
<td>Access grant identifier used to retrieve or revoke a specific grant. The value corresponds to the id field of the JSON array of grants returned from a previous GET /oauth/clients/{clientId}/grants or GET /oauth/users/{userKey}/grants. If this parameter is not specified, the request applies to all grants for the client or user.</td>
</tr>
</tbody>
</table>

**Cross Site Request Forgery Protection**

Both endpoints require the HTTP Header X-XSRF-HEADER with any value to prevent cross site request forgery.

**Example request and JSON response**

Request to retrieve all grants for client `im_client`:
GET /pf-ws/rest/oauth/clients/im_client/grants HTTP/1.1
Host: localhost:9031
Authorization: Basic YWRtaW46MkZlZGVyYXRl
X-XSRF-HEADER:PingFederate

Response could be:

```json
{
    "items": [
```
Session Revocation API

PingFederate includes a REST-based Web Service for Back-Channel Session Revocation. This service enables OAuth clients to add sessions to the revocation list or to query their revocation status. The Grant Access to Session Revocation API option must be selected in its client configuration (see “Back-Channel Session Revocation” on page 158).

Important: OAuth clients must authenticate to the Web Service using their Client Secret values via HTTP Basic authentication or Client Certificates (see “Configuring a Client” on page 178).

Endpoint: /pf-ws/rest/sessionMgmt/revokedSris

Tip: Information about the Session Revocation API endpoint is also available in the OpenID Connect metadata (see “OpenID Connect Metadata Endpoint” on page 551). Look for ping_revoked_sris_endpoint in the metadata.

This resource accepts the methods POST and GET. It also requires the X-XSRF-HEADER HTTP header with any value to prevent cross site request forgery.

Note: The POST method described in this section requires the parameter name/value pair formatted in JavaScript Object Notation (JSON).
POST

Description
A POST request adds a session to the revocation list based on its session identifier (id) in the POST data. The value of id corresponds to that of pi.sri in the ID Token (see “Configuring OpenID Connect Policies” on page 170). The required Content-Type is application/json.

Sample POST request and responses
A POST request to add a session with a session identifier of abc123 to the revocation list:

```plaintext
POST /pf-ws/rest/sessionMgmt/revokedSris
Host: localhost:9031
Authorization: Basic YWRtaW46MkZlZGVyYXRl
X-XSRF-HEADER: PingFederate
Content-Type:application/json

{"id":"abc123"}
```

Possible HTTP response status codes:
- 201 – Created
  The session is added to the revocation list.
- 400 – Bad Request
  The X-XSRF-HEADER HTTP header is not found in the HTTP POST request.
- 401 – Unauthorized
  The response contains details as to why the attempt failed.
- 415 – Unsupported Media Type
  The Content-Type: application/json HTTP header is not found in the HTTP POST request.
- 500 – Internal Server Error
  An unknown error has occurred.

GET

Description
A GET request sends a query for the revocation status for a session with its session identifier (id) appended to the endpoint. The value of id corresponds to that of pi.sri in the ID Token (see “Configuring a Client” on page 178).

Sample GET request and responses
A GET request to query the revocation status for a session with a session identifier of abc123:

```plaintext
GET /pf-ws/rest/sessionMgmt/revokedSris/abc123
Host: localhost:9031
Authorization: Basic YWRtaW46MkZlZGVyYXRl
X-XSRF-HEADER: PingFederate
```

Possible HTTP response status codes:
- 200 – OK
  ("id":"abc123") is found in the revocation list.
- 400 – Bad Request
  The X-XSRF-HEADER HTTP header is not found in the HTTP POST request.
• 401 – Unauthorized
   The response contains details as to why the attempt failed.
• 404 – Not Found
   
   {"resultId":"session_mgmt_sri_not_revoked","message":"The SRI has not been revoked."} if the session is not found in the revocation list.
• 500 – Internal Server Error
   An unknown error has occurred.

PingFederate Administrative API

PingFederate includes a REST-based application programming interface (API) for administrative functions. The administrative API provides a programmatic way to make configuration changes to PingFederate as an alternative to using the administrative console. The configuration changes that you can make through the administrative API include, but are not limited to:

• Server settings
• Connections
• Keys and Certificates
• OAuth settings
• Cluster Management

For a complete list, see “Using the API Interactive Documentation” on page 570.

PingFederate records actions performed via the administrative API in the admin-api.log file (see “Administrator API Audit Logging” on page 47).

The API can be called from an interactive user interface, custom applications, or from command line tools such as cURL.

---

**Important:** For known limitations, see Release Notes.

---

Using the API Interactive Documentation

PingFederate provides an interactive tool for use by both developers and non-developers to explore the API endpoints, view documentation for the API, and experiment with API calls.

**To access the interactive documentation in PingFederate:**

1. Start PingFederate.
2. Launch your browser and go to:
   
   https://<DNS_NAME>:9999/pf-admin-api/api-docs/
   
   where <DNS_NAME> is the fully qualified name of the machine running the PingFederate server.

**To test an administrative API using the interactive documentation:**

1. Click a section of the administrative API you would like to explore—for example, /sp/idpConnections.
2. Expand the method you want to use—for example, GET.
3. Enter any required parameters.
4. Click **Try it out!**


   **Note:** You may be prompted to log in using administrative credentials over HTTP Basic Authentication. Enter the credentials of any existing administrator. The role of the administrator affects their access to the requested API.
Using Attribute Mapping Expressions

PingFederate provides an advanced option allowing administrators to map user attributes by way of an expression language. Because the option carries with it a potential for misuse, however, it is disabled in the administrative console for security reasons.

**Tip:** If you are upgrading to PingFederate 5.1 or higher and importing a configuration archive that uses expression mapping, the feature is enabled automatically.

This appendix describes the option, which is based on the Object-Graph Navigation Language (OGNL), and how to enable or disable it.

**Caution:** The security concern posed by OGNL is related to a potential for abuse by PingFederate administrative users within an organization; the concern is not related to any known external threats. We recommend, however, that the option be enabled only if required.

## Enabling and Disabling Expressions

An administrator can manually enable or disable OGNL by editing a configuration file located in:

```
<pf_install>/pingfederate/server/default/data/config-store/
```

**Important:** If OGNL is enabled and expressions configured anywhere in the administrative console, disabling the feature causes errors during runtime processing.
To enable or disable OGNL expressions:
1. In the directory cited above, open the file:
   org.sourceid.common.ExpressionManager.xml
2. Change the value of the element named evaluate Expressions to either true or false and save the file. For example:
   
   ```xml
   <item name="evaluateExpressions">true</item>
   ```

   **Note:** The absence of a value (the installed default) does not necessarily disable the use of OGNL expressions. To facilitate backward compatibility, when no value is present, configuration archives containing expressions can be imported successfully, and further use of the feature is enabled. (The term “silent” is used for this condition in the server log.)

3. Start or restart PingFederate.

   **Tip:** If you are enabling OGNL to use for mapping Outbound-provisioning attributes, it is not necessary to restart the PingFederate server.

When you enable OGNL expressions, these expressions are available for use in multiple locations:

- Drop-down menus under Source in each of the administrative-console Fulfillment screens
- The provisioning attribute-mapping screen (when Outbound Provisioning itself is enabled—see “Outbound Provisioning for IdPs” on page 34)
- The Show Advanced Criteria section on the Issuance Criteria screen following each of the administrative-console Fulfillment screens

![Figure 5: Attribute Contract Fulfillment (Example)](image)

When you make this selection, you can enter the expression in the text field provided. You can also test expressions (see “Using the OGNL Edit Screen” on page 577).

**Constructing Expressions**

OGNL is based on the Java programming language. OGNL expressions are useful for evaluating and manipulating attribute values and returning information based on the
results. You can also transform a range of values into a text description or do the same for a sequence of ranges.

Use the # symbol to reference OGNL variables. For an IdP, PingFederate provides predefined OGNL variables for IdP-adapter attributes, any attributes retrieved from data stores, and attributes for token authorization. For an SP, variables are available for attributes received in an assertion, an attribute query, and attributes for token authorization. For example, the SAML_SUBJECT value may be retrieved using:

```
#SAML_SUBJECT
```

**Note:** Use the following construction for any attributes from any source that contain special characters (hyphens, for example), which cannot be parsed by OGNL:

```
this.get("<attribute_name>")
```

**Important:** Because OGNL uses the “at” symbol (@) to reference static Java methods, expressions containing the symbol must be enclosed in double quotes; otherwise, expression parsing fails. For example:

```
#SAML_SUBJECT="usr@msn.com"
not:
#SAML_SUBJECT=usr@msn.com
```

For more information, see “Using the OGNL Edit Screen” on page 577.

For more information about OGNL, including detailed user documentation, see the Apache Commons OGNL page (commons.apache.org/ognl).

### Data Store Syntax

For data-store attributes with an attribute source ID, use this syntax:

```
#this.get("ds.attr-source-id.attribute_name")
```

For data-store attributes without an attribute source ID, use this syntax:

```
#this.get("ds.attribute_name")
```

### Issuance Criteria Syntax

To access mapped attributes when configuring token-authorization expressions, use this syntax:

```
#this.get("mapped.attribute_name")
```

To access most context attributes when configuring token-authorization expressions, use this syntax:

```
#this.get("context.attribute_name")
```

To access the HTTP Request context attribute, use this syntax:

```
#this.get("context.HttpRequest").getObjectValue()
```

The returned value will be an instance of javax.servlet.http.HttpServletRequest (see http://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletRequest.html).
Expression Examples

Below are examples of using OGNL expressions for attribute mapping and token authorization.

**General**

In the expression below, the value of the attribute “net-worth” is transformed first to eliminate any dollar signs or commas, then the result is evaluated to determine whether the user’s net worth falls into a “bronze,” “silver,” or “gold” category:

#result=#this.get("net-worth").toString(),
#result=#result.replace("$",""),
#result=#result.replace("",""),
#result < 500000 ? "bronze" :
#result < 1000000 ? "silver" : "gold"

**Token Authorization**

The expression below verifies if a user is a member of the “Contractor” or “Employee” group:

#this.get("ds.ldap.memberOf")!=null?
#this.get("ds.ldap.memberOf").toString().matches(".*CN=Contractor,cn=users,dc=company,dc=com.*|
.*CN=Employee,cn=users,dc=company,dc=com.*")
@java.lang.Boolean@FALSE

**HTTP Request Context**

The example below may be used to retrieve a value from an HTTP request object, in this case the `User-Agent` header, for comparison to a value required for token authorization.

#this.get("context.HttpRequest")
.getObjectValue().getHeader("User-Agent")
.equals("somevalue")

**STS Client Authentication Context**

The STS SSL Client Certificate Chain example below checks that the issuer of the client certificate matches the specified DN.

#this.get("context.StsSSLClientCertChain").getObjectValue()[1].
getSubjectX500Principal().equals(new
javax.security.auth.x500.X500Principal("CN=Ping Identity
Engineering,OU=Engineering,O=Ping
Identity,L=Denver,ST=CO,C=USA"))

In the example above,
#this.get("context.StsSSLClientCertChain").getObjectValue()
returns an array of java.security.cert.X509Certificate instances (see http://
docs.oracle.com/javase/7/docs/api/java/security/cert/X509Certificate.html). This
array starts with the client certificate itself.

**Issuance Criteria and Multiple Virtual Server IDs**

When you use virtual server IDs to connect to multiple environments in one connection (see “Connecting to a Partner in Multiple Connections” on page 38),
Using the OGNL Edit Screen

verifying at runtime the virtual server ID in conjunction with other end-user attributes, such as group membership, protects against unauthorized access.

For instance, both the sales and the support departments of contoso.com (the IdP) have their own departmental subdomains, sales.contoso.com and support.contoso.com. The SP identifies both environments under the parent domain, contoso.com.

In this scenario, the PingFederate IdP server can be configured to include both sales.contoso.com and support.contoso.com as the virtual server IDs in the SP connection.

If you use one IdP adapter to authenticate end users from both departments, use an OGNL expression to cross-check the virtual server ID information in the request and the end user's group membership information. For example:

```
#this.get("ds.memberOf")!=null?
{
    #this.get("ds.memberOf").toString().matches("(?i)
    CN=Eng,OU=E,DC=contoso,DC=com")
    &&
    #this.get("context.VirtualServerId").toString()==
    "Engineering"
}||
{
    #this.get("ds.memberOf").toString().matches("(?i)
    CN=Mkt,OU=M,DC=contoso,DC=com")
    &&
    #this.get("context.VirtualServerId").toString()==
    "Marketing"
}
): false
```

**Note:** The line breaks above are for publication readability only; statements calling methods whose arguments are enclosed in quotes must be entered on a single line.

---

Using the OGNL Edit Screen

An in-line editor is available for OGNL expressions. The editor validates the expression and allows an administrator to enter input values and test the resulting output.

**Tip:** For information about using OGNL, refer to the Apache Commons OGNL page (commons.apache.org/ognl/index.html).

- To reach the OGNL editor, click **Edit** under Actions for an expression on any of the attribute Fulfillment screens or click **Test** in the Show Advanced Criteria section on the Issuance Criteria screen.

  The **Edit** action is also available on Issuance Criteria screens for expressions (see “About Token Authorization” on page 24).
### Using Attribute Mapping Expressions

**Note:** The test function does not work for the `context.httpRequest` attribute, because it’s value is an object rather than text.

Here is an example of the edit screen, from the IdP configuration flow:

![Edit Screen Example](image)

**To test an expression:**

1. Enter an input value in the **Value** text box associated with the attribute.
2. Click the **Test** link near the bottom right of the screen.
   
   If the expression contains no errors, the result appears under **Test Results**.

**Important:** If you make changes to an expression and want to save it, click **Update** under Actions. To discard changes, click the **Cancel** link under Actions; clicking the **Cancel** button near the bottom of the screen discards all changes you made in the current task.
Basic troubleshooting tips are provided here to help overcome common difficulties. Help is also available from the Support Center at pingidentity.com.

This appendix contains the following sections:
- “Data Store Issues” on page 581
- “Installation Issues” on page 582
- “Runtime Issues” on page 582
- “Server Startup” on page 582

## Data Store Issues

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>When setting up the JDBC data store, a connection cannot be established.</td>
<td>Verify that the proper drivers and connectors have been installed. Also, verify the connection URL, username, and password. If unsuccessful, contact your database administrator.</td>
</tr>
<tr>
<td>Cannot connect to a Directory Service with the LDAP protocol.</td>
<td>Verify the connection URL, port, principal, and credentials. If unsuccessful, contact your system administrator. If using LDAP with SSL/TLS (ldaps://), ensure the LDAP server’s SSL certificate is signed by a trusted certificate authority, or import the certificate into PingFederate (see “Trusted Certificate Authorities” on page 204).</td>
</tr>
</tbody>
</table>
Appendix I
Troubleshooting

Installation Issues

Table 26: Troubleshooting Installation

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error message “Not enough memory on the server”</td>
<td>Verify that there is at least 1,024 MB of RAM installed on the server (see “System Requirements” in the “Installation” chapter of Getting Started).</td>
</tr>
</tbody>
</table>

Runtime Issues

Table 27: Troubleshooting Runtime Issues

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificates unexpectedly expire.</td>
<td>Verify that the server clocks are synchronized on both sides of the federation. Note that you can configure PingFederate to notify administrators in advance of impending certificate expiration (see “Configuring Runtime Notifications” on page 99).</td>
</tr>
<tr>
<td>Receiving CrossModule/Network Errors</td>
<td>Verify network connections to the Hardware Security Modules (HSMs) are active and running. Also ensure the HSMs have not been unintentionally shut down.</td>
</tr>
</tbody>
</table>

Server Startup

Table 28: Troubleshooting the PingFederate Server

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PingFederate does not start.</td>
<td>Make sure that the Java SDK is installed (see “Installing the JDK” in the “Installation” chapter of Getting Started).</td>
</tr>
<tr>
<td>The server starts but indicates the license file is not found or invalid.</td>
<td>Ensure a current license is installed (see “Installing PingFederate” on page 16). If the server is part of a cluster, make sure the license is installed for only one server node, then restart all nodes.</td>
</tr>
</tbody>
</table>
Glossary

access token
A data object by which a client authenticates to a Resource Server and lays claim to authorizations for accessing particular resources.

account link
A persistent name identifier that enables federation of separately established accounts among disparate domains (see also account linking and pseudonym).

account linking
A form of identity mapping among separate user accounts managed under different domains. The mapping typically involves a name identifier—which may be a pseudonym—used to link the user to each account. The identifier is persisted at the SP site to enable seamless SSO/SLO. Additional attributes may be sent with the identifier.

account mapping
A form of identity mapping by which one or more user attributes is passed in a single sign-on transaction. The attributes are used at the destination site as a means identifying the user and looking up local account information.

adapter
Plug-in software that allows PingFederate to interact with Web applications and authentication systems (see “SSO Integration Kits and Adapters” on page 14).

adapter contract
A list of attributes “hard-wired” to an adapter and conveyed generally via cookies between the adapter and application.

artifact
A reference to a SAML protocol message. The federation partner that receives the artifact dereferences it, identifying the sender, and requests the complete message in a separate SOAP transaction.

Artifact Resolution Service
The SOAP endpoint that processes artifacts returned from a federation partner to retrieve the referenced XML message. Can be used to dereference authentication requests, assertion responses, and SLO messages.

assertion
A SAML XML document that contains identifying information about a particular subject; i.e., a person, company, application, or system. A SAML assertion can contain authentication, authorization, and attribute information about the subject.

Assertion Consumer Service
A SAML-compliant portion of PingFederate in an SP role that receives and processes assertions from an IdP.

attributes
Distinct characteristics that describe a subject. If the subject is a Web site user, attributes may include a name, group affiliation, email address, etc.
attribute contract
A list of attributes, agreed to by the partners in an identity federation, representing information about a user (SAML subject). The attributes are sent from the IdP to the SP during SSO or STS processing.

attribute mapping
A form of identity mapping between IdP and SP user accounts that uses attributes to identify the user or provide supplemental information.

attribute source
An data source used to fulfill a requestor's attribute contract.

audience
The XML element in a SAML assertion that uniquely identifies a Service Provider.

authentication context
An element in a SAML assertion indicating the method or process used by an IdP to authenticate the subject of the assertion; may be used for authorization decisions or auditing compliance.

attribute source
Specific database or directory location containing data needed by an IdP to fulfill a connection partner's attribute contract or by an SP to look up additional attributes to fulfill an adapter contract.

back-channel
Server-to-server, cross-domain communication path using a protocol, typically SOAP, that does not rely on a browser as an intermediary.

binding
A mapping of SAML request and response messages to specific transport protocols (redirect, POST, or artifact).

certificate
A digital file used for identity verification and other security purposes. The certificate, which is often issued by a Certificate Authority (CA), contains a public key, which can be used to verify the originator's identity.

Certificate Revocation List (CRL)
A list of revoked signing certificates, maintained by the issuing authority at a public URL.

channel
A dedicated Outbound Provisioning configuration specific to a particular service partner, data source, and target service.

connection partner
Entities, such as companies, that are part of an identity federation. These entities are referred to as connection partners in the PingFederate configuration process.

credential
Information used to identify a subject for access purposes (e.g., username and password). A credential can also be a certificate.

Database Management System
A system for storing and maintaining user account information and attributes. The tables and columns in the RDBMS are used by PingFederate to create user look-up and attribute retrieval queries. (See Java Database Connectivity.)
data store
A database or directory location containing user account records and associated user attributes.

Data Encryption Standard (DES)
A symmetric-key Standard.

defederation
Optional user-initiated delinking of an identity federation that uses a persistent name identifier or pseudonym for account linking.

digital signature
A process for verifying the identity of the originator of an electronic document and whether the document has been intercepted or altered. The process involves message signing, signature validation, and signing policy coordination between partners.

data store
A database or directory location containing user account records and associated user attributes.

endpoint
A terminal or gateway that generates or terminates a stream of information. For example, a PingFederate SP server contains an endpoint for the Assertion Consumer URL.

entity ID
The XML element in a SAML assertion that uniquely identifies an Identity Provider.

Extensible Markup Language
A structured, hierarchical text format—based on SGML (Standard Generalized Markup Language)—for the flexible and organized exchange of data.

grant type
The intermediate credentials that represent a resource owner authorization. Grant types are exchanged by the client with the OAuth Authorization Server in order to obtain an access token.
HTTP cookie
Information sent from a server to a Web browser to identify a registered Web site user. Once the cookie is placed in the browser, it is sent back to the server to identify the user every time the user accesses the site. PingFederate’s integration adapters interface with the cookie.

HTTP header
The section of an HTTP request or response containing information about the client or the server. PingFederate can use HTTP headers to look up session information passed by the IdP’s Web application.

HTTP request parameter
A named parameter sent as part of a URL request from a browser to a Web server.

identity federation
A trust agreement between or among organizations, implemented using accepted standards, to provide user-authentication tokens and other user or system attributes securely across domains, primarily to enable cross-domain SSO.

Identity Provider
The identity source or SAML authority that authenticates a subject and provides an SP with a security assertion vouching for that authentication.

IdP-initiated SSO or SLO
An identity federation transaction in which the initial action requiring a security context from an IdP occurs at an IdP’s site. For example, the user is logged on to the IdP and requests protected resources on an SP. The IdP sends authentication information to the SP.

inbound
A direction of message flow coming into a server relative to the server’s identity federation role (IdP or SP). For an IdP, inbound messages include SAML authentication requests. For an SP, inbound messages include SAML assertions.

Java Database Connectivity (JDBC)
A Java API that allows Java programs to interact with databases.

Kerberos ticket
The security token for the Kerberos protocol.

Key Distribution Center
The control center for authentication and authorization for Kerberos.

keysize
The length (in bits) of each key in a keypair.

keypair
The private key and public key represented by a certificate. PingFederate uses the private key of its keypair(s) to generate signatures for assertions, requests, and responses, as applicable.

Lightweight Directory Access Protocol
A set of protocols used for accessing information directories. PingFederate uses the LDAP v3 protocol for user look-up and attribute processing.

metadata
The SAML 2.0 standards define a metadata exchange schema for conveying XML-formatted information between two SAML entities. Metadata includes endpoint URLs, binding types, attributes, and security-policy information.

Network Access Server
(NAS) A RADIUS client server that provides a single point of access to a protected resource.

OAuth Authorization Server
A server that issues access tokens to clients (sometimes on behalf of a resource owner) for use in authenticating a subsequent Representational State Transfer (REST) API call.

OAuth Client
An application that desires access to a resource protected by a Resource Server and interacts with an OAuth Authorization Server to obtain access tokens to do so.

Online Certificate Status Protocol
(OCSP) A standard developed by the Internet Engineering Task Force that enables applications to obtain the current status of signing certificates, indicating whether a certificate has been revoked, via HTTP.

opaque
Not readable. If a user’s subject identifier is opaque, the an SSO partner cannot directly identify the user with reference to the source. An persistent identifier, or pseudonym, can be used for Account Linking.

outbound
A direction of message flow leaving a server. For an IdP, outbound messages include SAML assertions. For an SP, outbound messages include SAML authentication requests.

partner
See connection partner.
policy
A set of rules for handling security token requests in PingFederate.

portal
A Web-based application, accessed using a Web browser, that often aggregates content from multiple providers and/or serves as a central point of entry.

POST
An HTTP method of transmitting data contained in HTML forms, by which the data appears in the message body.

Primary Domain Controller
A role that is assigned to a particular server participating in a Windows network.

principal
A user, system, or process whose identity can be authenticated. See subject.

profiles
Rules that describe how to embed SAML assertions into and extract them out of other protocols in order to enable SSO or SLO. Profiles describe SAML request and response flows that fulfill specific use cases.

protected resource
Information, typically accessed via a Web URL, that is protected by an access management system. See target URL.

protocol
An agreed-upon format for transmitting data. XML format of SAML request or response messages.

pseudonym
A persistent name identifier assigned to a user and shared among entities, usually with the user's permission, to enable SSO and SLO. Pseudonyms are often used with the SAML account linking protocol to enable SSO while preventing the discovery of the user's identity or activities.

Public Key Infrastructure (PKI) Enables users of an unsecured public network, such as the Internet, to securely and privately exchange data and money through the use of keypairs and certificates. The PKI provides for a digital certificate that can identify an individual or an organization and directory services that can store and, when necessary, revoke the certificates.

redirect
A SAML binding that conveys a request or response by sending the user's browser to another location. For instance, an authentication request can be sent from an SP through a browser to an IdP.

refresh token
A long-lived token used by the client to obtain a new access token without having to obtain fresh authorization from the resource owner.

Remote Authentication Dial-in User Service (RADIUS) A networking protocol for user-access management that includes specifications for two-factor authentication.

<ResponseSecurityToken>
(RST) WS-Trust or WS-Federation XML element identifying a request for validation of a security token, or for validation and then issuance of a replacement security token.

<ResponseSecurityTokenResponse>
WS-Trust or WS-Federation XML element identifying a response to an RST and containing either the status of the submitted security token or both the status and (if requested and the received token is valid) a newly issued token for further SSO or Web-Services processing.

Resource Server
A server capable of accepting and responding to resource requests on which an access token is presented.

SAML
See Security Assertion Markup Language.

SAML authority
A security domain that issues SAML assertions.

scope
Permissions (for example, creating an event on a calendar) associated with an access token.

Secure Sockets Layer (SSL) An encryption protocol that sends data between a client and server over a secure HTTP connection.

Security Assertion Markup Language (SAML) A standard, XML-based, message-exchange framework enabling the secure transmittal of authentication tokens and other user attributes across domains.

System for Cross-domain Identity Management (SCIM) A REST-based protocol for provisioning and managing user identities across the Internet (see www.simplecloud.info).
**security domain**
An application or group of applications that trust a common security token used for authentication, authorization, or session management. The token is issued to a user after the user has authenticated to the security domain.

**security token**
A collection of information used to establish acceptable identity for security purposes. Tokens can be in binary or XML format. A SAML assertion is one kind of security token.

**Security Token Service**
An entity responsible for responding to WS-Trust requests for validation and issuance of security tokens used for SSO authentication to Web Services.

**service-oriented architecture**
A loosely coupled application architecture in which all functions or services are accessible via standard protocols. Interfaces are platform and programming-language independent.

**Service Provider**
A system entity that provides access to a protected resource based on authentication information supplied by an IdP.

**SP-initiated SSO or SLO**
An identity-federation transaction in which the initial action requiring a security context from an IdP occurs at a SP's site.

**session persistence**
A mechanism for identifying a user or browser for subsequent requests to a server, needed because the HTTP protocol is stateless. This information is used to lookup state information for the user—for example, items in a shopping cart. PingFederate does not implement session persistence; it facilitates the communication of session information between systems that do.

**Simple Object Access Protocol**
(SOAP) Defines the use of XML and HTTP to access services, objects, and servers in a platform-independent manner.

**Single Logout**
The process of logging a user out of multiple “session participants” or sites where the user has started an SSO session.

**Single Logout Return Service**
The SAML implementation endpoint URL that returns logout requests.

**Single Logout Service**
The SAML implementation endpoint URL that receives logout requests for processing.

**Single Sign-On**
(SSO) The process of authenticating an identity (signing on) at one Web site (usually with a user ID and password) and then accessing resources secured by other domains without re-authenticating.

**Single Sign-on Service**
The SAML implementation endpoint URL that receives authentication requests for processing.

**Source ID**
A 20-byte sequence used to determine an IdP’s identity.

**subject**
A person, computer system, or application. In the SAML context, assertions make statements about subjects. See principal.

**target URL**
The SP’s protected resource; the end destination of an SSO event. See protected resource.

**transient name identifier**
A temporary ID used to preserve user anonymity while facilitating account linking.

**token authorization**
A mechanism for evaluating attribute criteria available during a transaction to determine whether a user is authorized to access resources. A token in this instance can mean any type of security token—for example, SSO, session cookie, or OAuth token.

**token exchange**
The process by which a security token is exchanged for another security token.

**token translators**
An aggregate term for both token processors (used by the IdP PingFederate Security Token Service (STS) to handle different types of incoming security tokens) and token generators (used by the SP PingFederate STS to issue various types of tokens).

**Uniform Resource Identifier**
Identifies a Web resource with a string of characters conforming to a specified format.

**Uniform Resource Locator**
Identifies a resource according to its Internet location.
Glossary

**virtual server ID**
An optional unique identifier by which an identity federation deployment can be known to a specific connection partner.

**Web Services Security**
A standard mechanism for securing Web Service interactions, often by binding a security token to the Web Service request.

**Web Services**
Nonbrowser-based, loosely coupled applications that provide modular, programming-language-independent access to specific functions and data across the Internet, via XML and standard protocols.

**Web Service Client**
An entity that requests a Web Service interaction. In the context of an STS, the Web Service Client would request that a security token be issued for the interaction.

**Web Service Enhancement**
Supplemental software for the .NET framework provided by Microsoft.

**Web Service Provider**
In the context of an STS, an entity that requests validation of the security token sent with a client’s request for service.

**WS-SX**
The OASIS committee working on WS-Trust.

**WS-Trust**
A standard protocol by which an application can request that an STS issue, validate, or exchange security tokens.
# List of Acronyms

<table>
<thead>
<tr>
<th>ACS</th>
<th>Assertion Consumer Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programmer Interface</td>
</tr>
<tr>
<td>ARS</td>
<td>Artifact Resolution Service</td>
</tr>
<tr>
<td>CA</td>
<td>Certificate Authority</td>
</tr>
<tr>
<td>CRL</td>
<td>Certificate Revocation List</td>
</tr>
<tr>
<td>CSR</td>
<td>Certificate Signing Request</td>
</tr>
<tr>
<td>DBMS</td>
<td>Database Management System</td>
</tr>
<tr>
<td>DMZ</td>
<td>Demilitarized Zone</td>
</tr>
<tr>
<td>DN</td>
<td>Distinguished Name (certificate identifier)</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>EIM</td>
<td>Enterprise Identity Management</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HTTP</td>
<td>HyperText Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>Secure HyperText Transfer Protocol</td>
</tr>
<tr>
<td>IdM</td>
<td>Identity Management</td>
</tr>
<tr>
<td>IdP</td>
<td>Identity Provider</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>J2SDK</td>
<td>Java 2 Software Development Kit</td>
</tr>
<tr>
<td>JDBC</td>
<td>Java Database Connectivity (JDBC)</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>NAS</td>
<td>Network Access Server</td>
</tr>
<tr>
<td>O</td>
<td>Organization</td>
</tr>
<tr>
<td>OASIS</td>
<td>Organization for the Advancement of Structured Information Standards</td>
</tr>
<tr>
<td>OCSP</td>
<td>Online Certificate Status Protocol</td>
</tr>
<tr>
<td>OU</td>
<td>Organizational Unit</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial-in User Service</td>
</tr>
<tr>
<td>RDBMS</td>
<td>Relational Database Management System</td>
</tr>
<tr>
<td>RST</td>
<td>&lt;RequestSecurityToken&gt;</td>
</tr>
<tr>
<td>RSTR</td>
<td>&lt;RequestSecurityTokenResponse&gt;</td>
</tr>
<tr>
<td>SAML</td>
<td>Security Assertion Markup Language</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software as a Service</td>
</tr>
<tr>
<td>SCIM</td>
<td>System for Cross-domain Identity Management</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>SP</td>
<td>Service Provider</td>
</tr>
</tbody>
</table>
## List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLO</td>
<td>Single Logout</td>
</tr>
<tr>
<td>SOA</td>
<td>service-oriented architecture</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td>SSL/TLS</td>
<td>Secure Sockets Layer/Transport Level Security</td>
</tr>
<tr>
<td>SSO</td>
<td>Single Sign-On</td>
</tr>
<tr>
<td>SSTC</td>
<td>Security Services Technical Committee (of OASIS)</td>
</tr>
<tr>
<td>STS</td>
<td>Security Token Service</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>WCF</td>
<td>Windows Communication Foundation</td>
</tr>
<tr>
<td>WIF</td>
<td>Windows Identity Foundation</td>
</tr>
<tr>
<td>WSC</td>
<td>Web Service Client</td>
</tr>
<tr>
<td>WSP</td>
<td>Web Service Provider</td>
</tr>
<tr>
<td>WSS</td>
<td>Web Services Security</td>
</tr>
<tr>
<td>XASP</td>
<td>X.509 Attribute Sharing Profile</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>