

PingFederate[®]

Version 7.3



Guides

Getting Started	3
Administrator Manual	75
Server Clustering Guide	722
SSO Integration Overview	744
SDK Developer Guide	755
Upgrade Utility User Guide	780
Release Notes	789

PingFederate®

Version 7.3

Getting Started



© 2005-2015 Ping Identity® Corporation. All rights reserved.

PingFederate *Getting Started*
Version 7.3
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909
Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to the online documentation at documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: **January, 2015**.

Contents

	Preface	1
	About This Manual	1
	Overview	1
	Intended Audience	2
	Text Conventions	2
	Other Documentation	2
Chapter 1	Introduction	5
	About Identity Federation and SSO	5
	Service Providers and Identity Providers	5
	Identity Federation Hub	6
	Security Token Service	7
	OAuth Authorization Server	8
	User Account Management	8
	Enterprise Deployment Architecture	9
	Additional Features	9
Chapter 2	Installation	13
	System Requirements	14
	Installing the JDK	17
	Installing PingFederate	17
	Running PingFederate for the First Time	18
	Deployment Options	20
	Running PingFederate as a Service	22
	Uninstalling PingFederate	25
	Uninstalling Services	25
Chapter 3	Console Navigation	27
	Using the Main Menu	27
	Navigating the Administrative Console	30
	About Tasks and Steps	30

	Console Buttons	31
Chapter 4	Supported Standards	33
	Federation Roles	33
	Terminology	34
	Browser-based SSO	36
	SAML 1.x Profiles	36
	SSO--Browser-Post	36
	SSO--Browser-Artifact	37
	SP-Initiated (“Destination-First”) SSO	38
	SAML 2.0 Profiles	39
	Single Sign-on	39
	Single Logout	50
	Attribute Query and XASP	50
	IdP Discovery	51
	WS-Federation	51
	Passive Requestor Profile	52
	About Account Linking	53
	Web Services Standards	54
	Web Services Security	55
	WS-Trust	55
	Request Types	56
	OAuth 2.0	57
	Web Redirect Flow	58
	SAML 2.0 Profile for OAuth 2.0 Authorization Grants	59
	OpenID Connect Support	60
	System for Cross-domain Identity Management (SCIM)	60
	Transport and Message Security	60
Appendix A	Using the Thales nShield Connect HSM	61
	HSM Operational Notes	61
	nShield Installation and Configuration	62
	Additional Steps for Server Clusters	64
Appendix B	Using the SafeNet Luna HSM	65
	Operational Notes	65
	Installation and Configuration	66

Preface

About This Manual

This guide provides information about getting started with Ping Identity's PingFederate to deploy a secure cloud-identity platform, including single sign-on (SSO) based on the latest security and e-business standards.

Overview

This document consists of:

- [Chapter 1 “Introduction”](#) — A high-level view of federated identity, secure Web SSO, and PingFederate features.
- [Chapter 2, “Installation”](#) — How to install PingFederate and run the administrative console for the first time.
- [Chapter 3 “Console Navigation”](#) — A primer on using the administrative console and configuration screens.
- [Chapter 4, “Supported Standards”](#) — An overview of industry standards that PingFederate supports, including the Security Assertion Markup Language (SAML) and WS-Federation.
- [Appendix A “Using the Thales nShield Connect HSM”](#) — How to install and configure PingFederate with the Thales nShield Connect Hardware Security Module as part of compliance with the Federal Information Processing Standard (FIPS) 140-2.
- [Appendix B, “Using the SafeNet Luna HSM”](#) — How to install and configure PingFederate with the SafeNet Luna Hardware Security Module as part of compliance with the Federal Information Processing Standard (FIPS) 140-2.

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.

Text Conventions

This document uses the text conventions identified below.

Table 1: Text Conventions

Convention	Description
Fixed Width	Indicates text that must be typed exactly as shown in the instructions. Also used to represent program code, file names, and directory paths.
Blue text	Indicates hypertext links.
<i>Italic</i>	Used for emphasis and document titles.
▶ [text]	Used for procedures where only one step is required.
Sans serif	Identifies descriptive text on a user-interface screen. Example: “Print Document dialog”
Sans serif bold	Identifies menu items, navigational links, or buttons. For example: Click Save .

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.

Administrator’s Manual – Provides key concepts as well as detailed instructions for using the PingFederate administrative console—also connection-endpoint and other Web-application developer information, a glossary, and a list of common acronyms.

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](#).

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.

Web Resources – Ping Identity continually updates its [Resource Center](http://www.pingidentity.com/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](http://www.pingidentity.com/support) (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](http://www.pingidentity.com/support) (www.pingidentity.com/support) if you encounter a problem.

Introduction

Welcome to PingFederate, Ping Identity's enterprise identity bridge. PingFederate enables outbound and inbound solutions for single sign-on (SSO), federated identity management, mobile identity security, API security, and social identity integration. Browser-based SSO extends employee, customer and partner identities across domains without passwords, using only standard identity protocols (Security Assertion Markup Language—SAML, WS-Federation, WS-Trust, and OAuth).

About Identity Federation and SSO

Federated identity management (or “identity federation”) enables enterprises to exchange identity information securely across domains, providing browser-based SSO. Federation is also used to integrate access to applications across distinct business units within a single organization. As organizations grow through acquisitions, or when business units maintain separate user repositories and authentication mechanisms across applications, a federated solution to browser-based SSO is desirable.

This cross-domain, identity-management solution provides numerous benefits, ranging from increased end-user satisfaction and enhanced customer relations to reduced cost and greater security and accountability.

For complete information about identity federation and the standards that support it, see [“Supported Standards”](#) on page 33.

Service Providers and Identity Providers

Identity federation standards identify two operational roles in an SSO transaction: the *identity provider* (IdP) and the *service provider* (SP). An IdP, for example, might be an enterprise that manages accounts for a large number of users who may need secure access to the Web-based applications or services of

customers, suppliers, and business partners. An SP might be a SaaS provider or a business-process outsourcing (BPO) vendor wanting to simplify client access to its services.

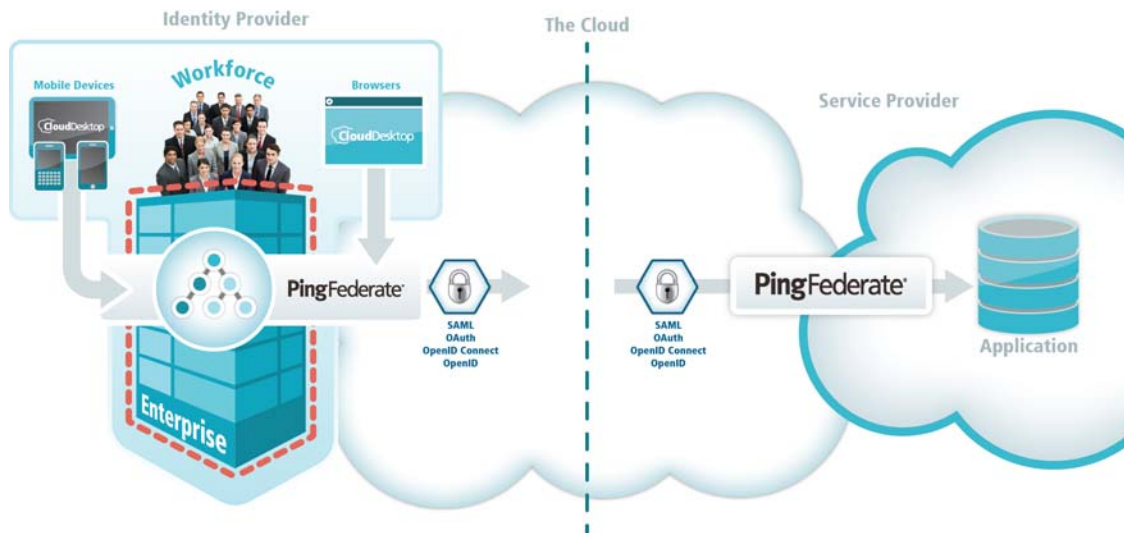


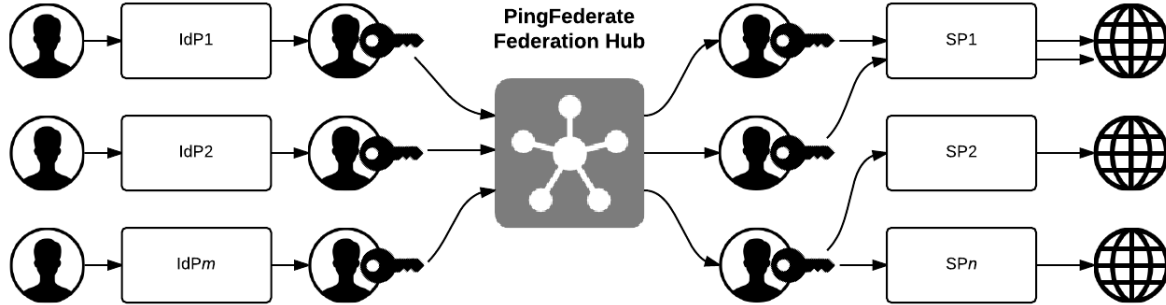
Figure 1: Secure Single Sign-on

Identity federation allows both types of organizations to define a trust relationship whereby the SP provides access to users from the IdP. The IdP continues to manage its users, and the SP trusts the IdP to authenticate them.

PingFederate provides complete support for both roles. Note that business processes of a single organization might encompass both SP and IdP use cases; this scenario can be handled by a single instance of PingFederate.

Identity Federation Hub

To most organizations, identity federation means negotiating and managing federation settings with partners. As the number of partners grows, so does the administrative overhead. In addition, different federation protocols may also hinder application development and SSO implementation. To remove these obstacles, PingFederate can be configured as a Federation Hub to extend federated access across partners supporting different federation standards, SAML and WS-Federation for example, as well as to provide a centralized console to simplify SSO administration. By bridging the identity providers and service providers through the federation hub, administrators also have the option to multiplex a single connection for multiple partners, adding additional use cases and reducing administration and implementation costs.



For more information, see [“Federation Hub”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.

Security Token Service

The PingFederate WS-Trust Security Token Service (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”](#) on page 54.)

The STS shares the core functionality of PingFederate, including console administration, identity and attribute mapping, and certificate security management. With PingFederate, Web Services can securely identify the end user who has initiated a transaction across domains, providing enhanced service while simultaneously ensuring appropriate information access and regulatory accountability.

PingFederate can be used in many different scenarios to address different identity and security problems as they relate to Web Services, service-oriented architecture (SOA), and Enterprise Service Buses. All of these scenarios share a recommended architectural approach that uses a SAML assertion as the standard security token shared between security domains. (For more information, see [“About WS-Trust STS”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.)

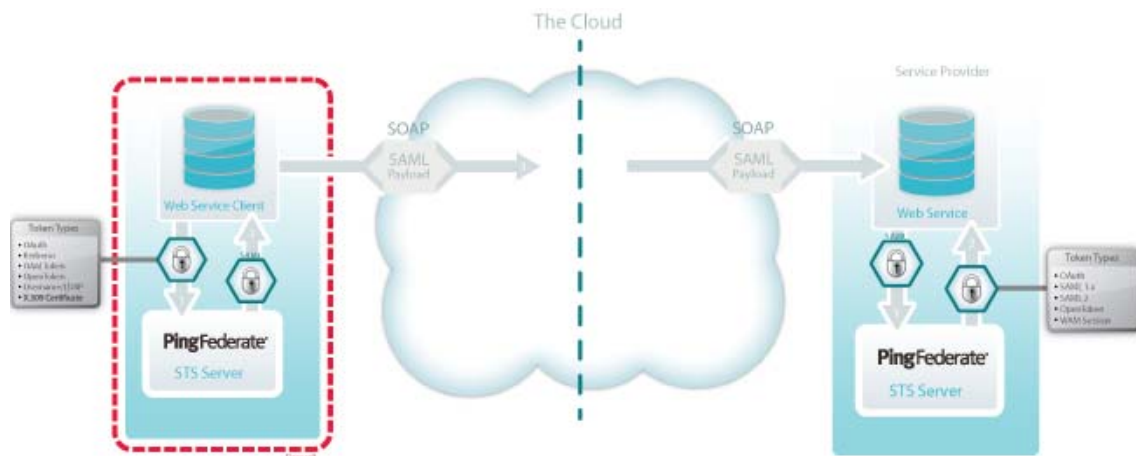


Figure 2: Security Token Service SSO

OAuth Authorization Server

PingFederate can act as an OAuth Authorization Server (OAuth AS), allowing a resource owner to grant authorization to a client requesting access to resources protected by a Resource Server. The OAuth AS issues tokens to clients on behalf of a resource for use in authenticating a subsequent API call—typically, but not exclusively a Representational State Transfer (REST) API. The PingFederate OAuth AS issues tokens to clients in several different scenarios, including:

- A web application wants access to a protected resource associated with a user and needs the user’s consent.
- A native application client on a mobile device or tablet wants to connect to a user’s online account and needs the user’s consent.
- An enterprise application client wants to access a protected resource hosted by a business partner, customer, or SaaS provider.

(For information about OAuth and the role of an AS, see “[OAuth 2.0](#)” on page 57.)

The PingFederate OAuth AS can be configured independently or in conjunction with STS and browser-based SSO for either an IdP or an SP deployment. For more information, see “[About OAuth](#)” in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.



Note: OAuth AS capabilities may require additional licenses. For more information, please contact sales@pingidentity.com.

User Account Management

In an identity federation, accounts are maintained for users at the IdP site. However, an SP will often have its own set of user accounts, some of which may correspond to IdP users. The SP may also need to establish and maintain parallel accounts for remote SSO users to enforce authorization policy, customize user experience, comply with regulations, or a combination of such purposes.

To facilitate cross-domain account management, PingFederate provides two kinds of user provisioning for browser-based SSO, one designed for an IdP and one for an SP:

- At an IdP site, an administrator can automatically provision and maintain user accounts for partner SPs who have implemented the System for Cross-domain Identity Management ([SCIM](#)) or, when optional plug-in SaaS Connectors are used, for selected hosted-software providers.
- At an SP site, an administrator can provision accounts within the organization automatically from SCIM-enable IdPs or use information from SAML assertions received during SSO events.

For more information, see [“User Provisioning”](#) in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.

Enterprise Deployment Architecture

With PingFederate’s enterprise-deployment architecture, all protocol definitions, public key infrastructure (PKI) keys, policies, profiles, etc., are managed in a single location, eliminating the need to maintain redundant copies of these configurations and trust relationships. Furthermore, when new protocols, profiles, or use cases need to be added, you only have to configure them once to make them available to your entire organization.

PingFederate also improves security by creating a single “doorway” in your perimeter through which all identity information must travel. Using PingFederate, all of your internal users who sign on to external applications exit through this doorway, while all external users who sign on to your internal systems enter through the same doorway.

The single-doorway approach also provides 100 percent visibility to all federation activities. The extensive auditing and logging capabilities of PingFederate enable you to satisfy all of your logging-related compliance and service-level requirements from a single location, as opposed to having to acquire and consolidate disparate logs from throughout your organization.

Use Case Configuration

By providing a single configuration paradigm supporting different protocols, PingFederate reduces complexity and learning curves. Furthermore, the step-by-step administrative console minimizes the potential for errors by guiding administrators through configuration steps applicable only to the business use cases they need to support.



Tip: For IdPs, connection templates that automatically configure many steps in the administrative console are available for several use cases, including setting up SSO connections to selected SaaS vendors. (For more information, see [“Outbound Provisioning for IdPs”](#) in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.)

Additional Features

PingFederate’s lightweight, stand-alone architecture means you can receive the benefits of standards-based SSO and API security integration without the cost and complexity of deploying a complete identity management (IdM) system. The PingFederate server integrates and coexists with existing home-grown and commercial IdM systems and applications, using these key features available separately from Ping Identity:

- [Integration Kits](#) – These tailored kits simplify integration with existing applications while minimizing impacts on existing infrastructure.

- **Token Translators** – These specialized plug-ins connect the STS with Web Service Providers and Clients to enable access to identity-enabled Web Services, which may require a range of different token types.
- **SaaS Connectors** – These plug-ins provide quick-connection templates and automated user provisioning and deprovisioning for selected SaaS providers, including Salesforce and Google Apps.
- **Cloud Identity Connectors** – These plug-ins allow cloud identity providers such as Facebook, Yahoo!, Google and Salesforce to authenticate and connect users to SSO-enabled applications.

Integration Kits

PingFederate provides a suite of integration kits to complete the first- and last-mile integration with your existing IdM systems and Web applications. PingFederate integration kits are available for download from the [Ping Identity Web site](#), take only minutes to install, and are configured from within the PingFederate administrative console.

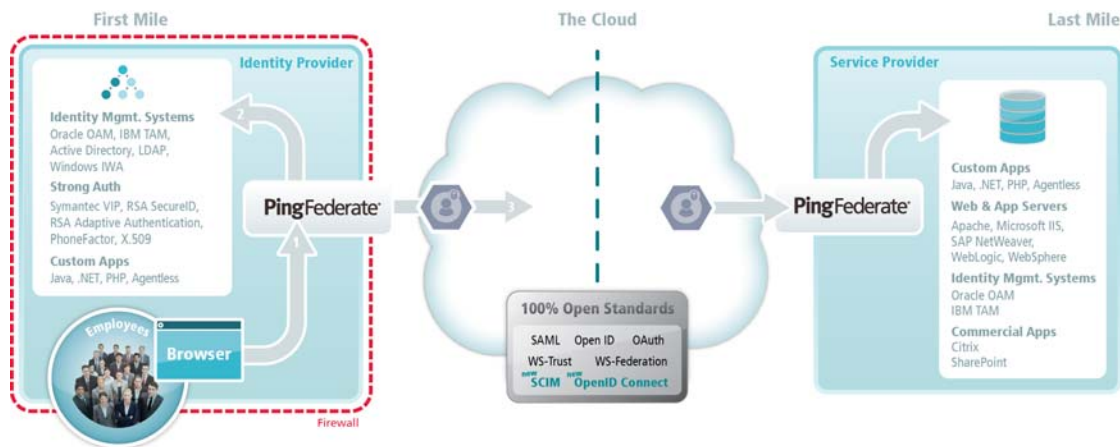


Figure 3: Multiple security-domain, multi-protocol federation

Integration kits enable rapid session integration with both existing authentication services and target applications. In addition, PingFederate includes a Software Development Kit for creating custom integrations.

For more information, see [“SSO Integration Kits and Adapters”](#) in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.

Token Translators

Ping Identity offers special Token Processors (for an IdP) and Token Generators (for an SP) to enable the WS-Trust STS to validate and issue a variety of token types. These plug-ins, which supplement built-in SAML token processing and generation, are designed to handle local identity tokens required in a variety of security contexts.

For more information, see [“Token Processors and Generators”](#) in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.

SaaS Connectors

SaaS Connectors offer a streamlined approach for browser-based SSO to selected SaaS providers—including automatic user provisioning and deprovisioning (see “[Outbound Provisioning for IdPs](#)” in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*). The Connector packages (available separately) include quick-connection templates, which automatically configure endpoints and other connection information for each provider.

Cloud Identity Connectors

Ping Identity offers social identity integration with social networking sites. The OpenID cloud-identity connector leverages OpenID 2.0 social networking providers (including Google and Yahoo!) for registration and access to cloud-based applications. Connectors for Twitter, LinkedIn, Windows Live, and Facebook leverage user logins for registration and access to cloud-based applications.

About PingOne

[PingOne](#) is Ping Identity's multi-tenant, identity-as-a-service solution. PingOne enables browser-based SSO and user provisioning for Identity Providers, and provides application providers with a rapid-deployment SSO capability. PingOne can be used together with PingFederate to provide a powerful solution coupling the benefits of an on-premise deployment with the flexibility of a cloud solution. For more information on PingOne, please visit <http://www.pingone.com>.

Installation

PingFederate is packaged as a stand-alone server based on J2EE application server technology.

This chapter covers:

- [“System Requirements”](#) on page 14
- [“Installing the JDK”](#) on page 17
- [“Installing PingFederate”](#) on page 17
- [“Running PingFederate for the First Time”](#) on page 18
- [“Deployment Options”](#) on page 20
- [“Running PingFederate as a Service”](#) on page 22
- [“Uninstalling PingFederate”](#) on page 25

System Requirements

PingFederate is certified as compatible for deployment and configuration with the minimum system specifications defined in the following sections.



Note: Ping Identity has qualified the following configurations and certified that they are compatible with the product. Variations of these platforms (for example, differences in operating system version or service pack) are supported up until the point at which an issue is suspected as being caused by the platform or other required software.

Operating Systems



Note: PingFederate has been tested with default configurations of operating system components. If your organization has customized implementations or has installed third-party plug-ins, deployment of the PingFederate server may be affected.

- Microsoft Windows Server 2008 R2 SP1
- Microsoft Windows Server 2012 Standard
- Microsoft Windows Server 2012 R2 Standard
- Oracle Enterprise Linux 6.5 (Red Hat Compatible Kernel)
- Oracle Solaris 10
- Red Hat Enterprise Linux ES 6.5
- Red Hat Enterprise Linux ES 7.0
- SUSE Linux Enterprise 11 SP3

Virtual Systems

Although Ping Identity does not qualify or recommend any specific virtual-machine (VM) products, PingFederate has been shown to run well on several, including VMWare, Xen, and Windows Hyper-V.



Note: This list of products is provided for example purposes only. We view all products in this category equally. Ping Identity accepts no responsibility for the performance of any specific virtualization software and in no way guarantees the performance and/or interoperability of any VM software with its products.

Java Environment

- Oracle Java SE Development Kit (JDK) 7 update 75 (64-bit)
- Oracle Java SE Development Kit (JDK) 8 update 31 (64-bit)

Supported Browsers for End Users

- Chrome
- Firefox
- Internet Explorer (version 9 and higher)

- Safari

Supported Browsers for the Administrative Console

- Chrome
- Firefox
- Internet Explorer (version 9 and higher).

Data Store Integration

For User-Attribute Lookup:

- Microsoft Active Directory (2008 R2 and 2012)
- Oracle Directory Server Enterprise Edition 11g
- Microsoft SQL Server (2012 and 2014)
- Oracle Database (10g and 11g R2)
- Oracle MySQL 5.6

For Outbound Provisioning:

- Provisioning Channel Data Source
 - Microsoft Active Directory (2008 R2 and 2012)
 - Oracle Directory Server Enterprise Edition 11g
- Provisioning Internal Data Store
 - Microsoft SQL Server (2012 and 2014)
 - Oracle Database 11g R2
 - Oracle MySQL 5.6

For Inbound Provisioning:

- Microsoft Active Directory (2008 R2 and 2012)

For Just-in-Time Provisioning (External Target Database):

- Microsoft SQL Server (2012 and 2014)

For Account Linking:

- Microsoft Active Directory (2008 R2 and 2012)
- Oracle Directory Server Enterprise Edition 11g
- Microsoft SQL Server (2012 and 2014)
- Oracle Database 11g R2
- Oracle MySQL 5.6

For OAuth Persistent Grants and Client Configuration:

- Microsoft SQL Server (2012 and 2014)
- Oracle Database 11g R2
- Oracle MySQL 5.6

Hardware Security Module (Optional)

- SafeNet Luna SA version 5.3

Client Driver Version: 5.3

Note: Luna SA does not support Java 8. Install Oracle JDK 7 on the PingFederate server (see “[Java Environment](#)” on page 14). For more information about Luna SA, see “[Using the SafeNet Luna HSM](#)” on page 65.

- Thales nShield Connect version 2.51.10

Client Driver Version: 11.70

Note: nShield Connect does not support Java 8. Install Oracle JDK 7 on the PingFederate server (see “[Java Environment](#)” on page 14). For more information about nShield Connect, see “[Using the Thales nShield Connect HSM](#)” on page 61.

Hardware Requirements



Note: Although it is possible to run PingFederate on less powerful hardware, the following guidelines accommodate disk space for default logging and auditing profiles and CPU resources for a moderate level of concurrent request processing.

Minimum Hardware Requirements

- Intel Pentium 4, 1.8 GHz processor
- 1 GB of RAM
- 250 MB of available hard drive space

Minimum Hardware Recommendations

- Multi-core Intel Xeon processor or higher
 - 4 CPU/Cores recommended
- Multi-core SPARC processor (Solaris)
 - 4 CPU/Cores recommended
- 4 GB of RAM
 - 1.5 GB available to PingFederate
- 500 MB of available hard drive space

Installing the JDK

You must install the Oracle JDK before running PingFederate, see “[Java Environment](#)” for more information.



Tip: Due to import control restrictions, the standard JDK distribution supports strong but not unlimited encryption. Stronger encryption is optional in several PingFederate and plug-in configurations. To use the strongest encryption, when permissible, after installing the JDK, download and install the appropriate version of “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” from the [Oracle download Web site](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (www.oracle.com/technetwork/java/javase/downloads/index.html).

For an example configuration where stronger encryption may be desirable, see “[Selecting an Encryption Certificate \(SAML\)](#)” in the “Identity Provider SSO Configuration” chapter of the PingFederate *Administrator’s Manual*.

To install the Oracle JDK for Windows and Linux:

1. Download and install the Oracle JDK from: www.oracle.com/technetwork/java/javase/downloads.
2. Set the `JAVA_HOME` environment variable to the JDK installation directory path and add the `/bin` directory to the `PATH` variable for your platform. Set the variables at either the system or user level.



Note: If you are running PingFederate as a service, you must set `JAVA_HOME` at the system level.

Installing PingFederate

You install PingFederate by extracting the distribution ZIP file.



Note: If your site requires compliance with FIPS 140-2, see either “[Using the Thales nShield Connect HSM](#)” on page 61 or “[Using the SafeNet Luna HSM](#)” on page 65 for additional installation information.



Important: On Unix or Linux you must install and run PingFederate under a local user account.

To install PingFederate:

1. Ensure you are logged on to your system with appropriate privileges to install and run an application.
2. Verify that the JDK is installed and that environment and `PATH` variables are set correctly (see “[Installing the JDK](#)” on page 17).

3. Extract the distribution ZIP file into an installation directory.



Note: To avoid future problems with automated upgrades, do *not* rename the installed `pingfederate` subdirectory. If you are installing multiple instances of PingFederate on the same machine (for example, in certain server-clustering scenarios), either install each instance in a different location or rename the higher level directory (`pingfederate-6.x.x`) to install a parallel file structure in the same location.

4. Request a license key via the [Ping Identity licensing Web page](http://www.pingidentity.com/support-and-downloads/licensing.cfm) (www.pingidentity.com/support-and-downloads/licensing.cfm).
5. Save the license key file in the directory:

```
<pf_install>/pingfederate/server/default/conf
```

Ensure the file is named:

```
pingfederate.lic
```



Tip: Alternatively, you can start PingFederate and import and validate the license prior to logging on (see [“Running PingFederate for the First Time”](#) on page 18).



Note: If you are deploying PingFederate in a cluster configuration, install the license key *only* on the administrative-console server—the license is retrieved by other servers automatically. (For more information about clustering, see the PingFederate Server Clustering Guide.)

Running PingFederate for the First Time

The first time you run the PingFederate administrative console, if you have not already installed a license from Ping Identity, you are asked to import it. After a license is installed, log on with the default username and password supplied with the distribution.



Tip: Later, depending on your network configuration and requirements, you can set up alternative means of console authentication (see [“Alternative Console Authentication”](#) in the “System Administration” chapter of the PingFederate *Administrator’s Manual*).

After launching the administrative console and logging on, you must change the default password. After that, “welcome” screens guide you through an initial setup process.

When the initial installation process is complete, the Main Menu opens (see [“Using the Main Menu”](#) on page 27).



Note: You can change the installation setup via menu choices under My Server on the Main Menu (see [“Managing Server Settings”](#) in the “System Settings” chapter of the *PingFederate Administrator’s Manual*).

To run PingFederate for the first time:

1. Start the PingFederate server by running the following script:

(Windows) <pf_install>/pingfederate/bin/run.bat

(Unix/Linux) <pf_install>/pingfederate/bin/run.sh

Wait for the script to finish the startup—the server is deployed when this message appears near the end of the sequence:

```
PingFederate started in [xx]s:[yy]ms
```

2. If you have not yet installed a PingFederate license, on the Import License screen locate and import the license file needed for this instance of PingFederate.

The license is validated and the file renamed correctly (if necessary) during the process. (For more information, see [“Installing a New License Key”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*.)

3. Launch your browser and go to:

```
https://<DNS_NAME>:9999/pingfederate/app
```

where <DNS_NAME> is the fully qualified name of the machine running the PingFederate server.



Note: The port number 9999 is set by default. For information on changing this setting, see [“Changing Configuration Parameters”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*.

4. Enter Username and Password.

Username: Administrator

Password: 2Federate

5. Change your password on the Change Password screen and click **Save**.



Note: The new password must be at least six characters and contain at least one uppercase, one lowercase, and one numeric character.



Important: Take steps to ensure that you do not forget the new password. For more information about passwords and user management, see [“Account Management”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*.

6. Complete the steps in the Configuring My Server screens.

For more information, refer to the context-sensitive **Help** pages or see sections under [“Managing Server Settings”](#) in the “System Settings” chapter of the *PingFederate Administrator’s Manual*.

Deployment Options

There are many options for deploying PingFederate in your network environment, depending on your needs and infrastructure capabilities.

For example, you can choose a stand-alone or proxy configuration, as described in this section. Or you can deploy multiple PingFederate servers in a cluster configuration for high availability, server redundancy, and failover recovery (see the *PingFederate Server Clustering Guide*).

Figure 4 illustrates PingFederate installed in the DMZ:

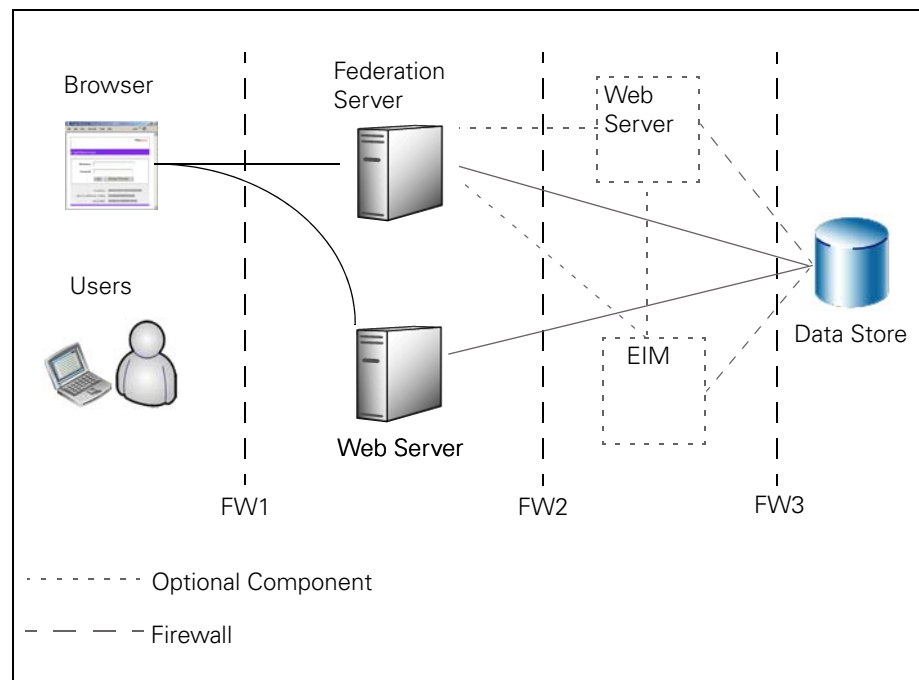


Figure 4: Stand-alone Deployment Example

In this configuration, users access PingFederate via a Web application server (and/or an Enterprise Identity Management system). PingFederate may, in turn, retrieve information from a data store to use in processing the transaction.

You can also deploy PingFederate with a proxy server. Figure 5 depicts a proxy-server configuration in which the proxy is accessed by users and Web browsers. The proxy, in turn, communicates with PingFederate to request SSO.

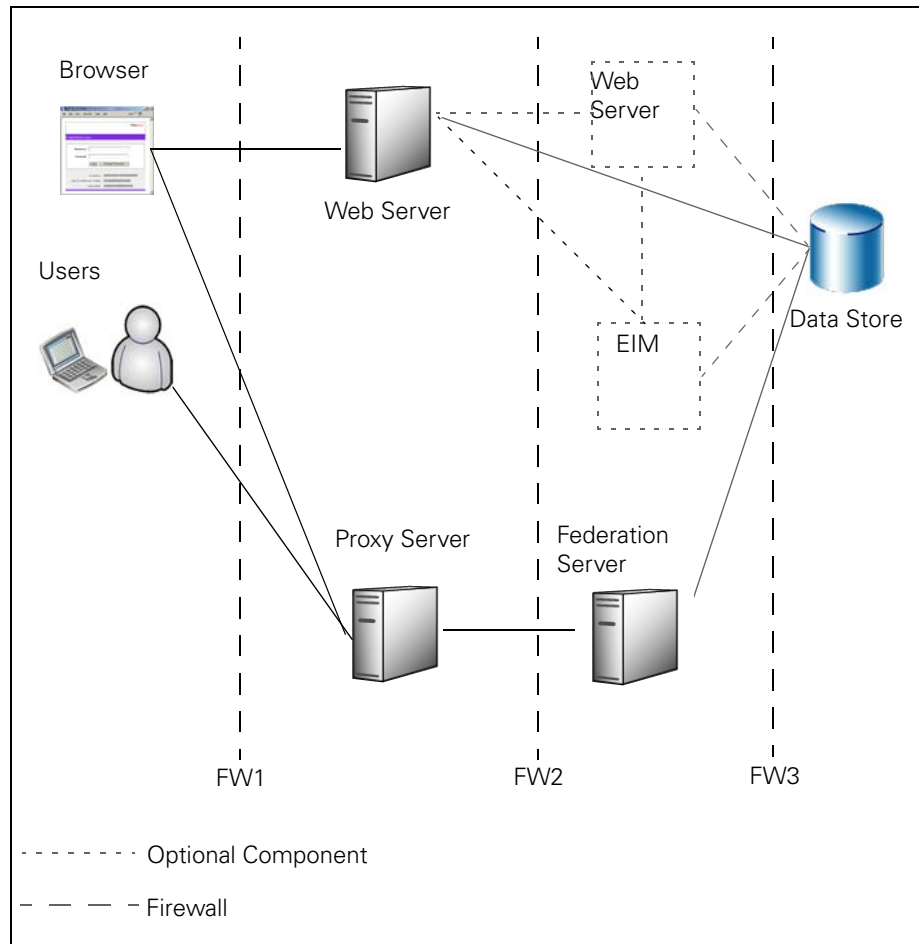


Figure 5: Proxy Deployment Example

Running PingFederate as a Service

You can set up PingFederate to run in the background as a service on either Windows or Linux.



Note: Before performing this procedure, ensure that PingFederate runs normally by manually starting the server (see [“Running PingFederate for the First Time”](#) on page 18).

Important: For Linux, when you start the server manually, you must run the startup script under the same user account that the service will use (see the procedure under [“\(Linux\)”](#) on page 23).

(Windows)

This installation enables PingFederate to start automatically when Windows is started or rebooted.



Note: If you are upgrading to a 64-bit service, you must first uninstall the previous PingFederate service (see [“Uninstalling Services”](#) on page 25).

To run PingFederate as a Windows service:

1. Complete the steps under [“Installing PingFederate”](#) on page 17.



Note: Ensure `JAVA_HOME` and `PATH` are set as system variables (see [“Installing the JDK”](#) on page 17).

2. Ensure you are logged on with full Administrator privileges.
3. Start PowerShell or Command Prompt as an Administrator.
4. In PowerShell or Command Prompt, run the `install-service.bat` file to install the service on one of the following platforms:

On a 64-bit Windows x86 platform, run `install-service.bat` from the directory:

```
<pf_install>\pingfederate\sbin\win-x86-64
```

Or:

On a 64-bit Windows Itanium platform, run `install-service.bat` from the directory:

```
<pf_install>\pingfederate\sbin\win-itanium-64
```

5. Access the Windows **Control Panel** > **Administrative Tools** > **Services**.
6. Right-click PingFederate Service from the list of available services and select **Start**.

The service starts immediately and will restart automatically on reboot. (You can change the default Start type setting in the **Properties** dialog.)

(Linux)

To run PingFederate as a service on Linux, you must place a script in the system initialization directory.



Note: If you are not using RedHat, you may need to modify references to the system initialization directory in this procedure—for example, Debian uses `/etc/init.d/` instead of `/etc/rc.d/init.d/`.

To run PingFederate as a Linux service (RedHat):

1. Complete the steps under “[Installing PingFederate](#)” on page 17.
2. Log on as root.
3. Create a new user account for the service.

For this procedure, the variable `<pf_user>` is used to refer to this account.



Note: Ensure the environment variable `JAVA_HOME` is set and the `PATH` variable updated for `<pf_user>` (see “[Installing the JDK](#)” on page 17).

4. Change the PingFederate installation directory (`<pf_install>`) ownership and ensure its read/write property:

```
chown -R <pf_user> <pf_install>
chmod -R 775 <pf_install>
```

5. Place the code below into a file called `<pf_user>` in the directory: `/etc/rc.d/init.d/`



Note: Replace instances of `<pf_user>` and `<pf_install>` in the script below, and in the commands that follow, with their respective values.

```
#!/bin/sh

start(){
    echo "starting PingFederate.."
    su - <pf_user> \
    -c '<pf_install>/pingfederate/sbin/pingfederate-run.sh \
    > /dev/null 2> /dev/null'
}

stop(){
    echo "stopping PingFederate.."
    su - <pf_user> \
    -c '<pf_install>/pingfederate/sbin/pingfederate-
shutdown.sh'
}
```

```
restart(){
    stop
    # padding time to stop before restart
    sleep 60
    # To protect against any services that are not stopped,
    # uncomment the following command.
    # (Warning: this kills all Java instances running as
    # <pf_user>.)
    # su - <pf_user> -c 'killall java'
    start
}
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        restart
        ;;
    *)
        echo "Usage: <pf_user> {start|stop|restart}"
        exit 1
esac
exit 0
```

6. Create symbolic links using commands listed below.

The links specify the order in which the PingFederate Server starts and stops.

```
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc3.d/S84<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc5.d/S84<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc4.d/S84<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc6.d/K15<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc0.d/K15<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc1.d/K15<pf_user>
ln -s /etc/rc.d/init.d/<pf_user> /etc/rc2.d/K15<pf_user>
```

7. Make the script executable (as root):

```
chmod 755 /etc/rc.d/init.d/<pf_user>
```

8. Test the script by entering:

```
service <pf_user> start
```

and then:

```
service <pf_user> stop
```

9. To start the service, enter:

```
service <pf_user> start
```

Uninstalling PingFederate

To uninstall PingFederate (on Windows or Linux):

1. If PingFederate is installed as a service, follow the platform-specific procedure in the next section, [“Uninstalling Services”](#).
2. Delete the PingFederate installation directory.

Uninstalling Services

(Windows)

To uninstall PingFederate as a Windows Service:

1. Access the **Windows Control Panel > Administrative Tools** and double-click **Services**.
2. Right-click PingFederate or PingFederate Service from the list of available services and select **Properties**.
3. Click **Stop** under the General tab in the Properties dialog window.
4. Run `uninstall-service.bat` from the `<pf_install>\pingfederate\sbin` subdirectory that corresponds to your platform processor.

(Linux)

To uninstall PingFederate as a Linux Service:

1. Log on as root.
2. Stop the service with the command:

```
service <pf_user> stop
```

where `<pf_user>` is the PingFederate service user account (see [“Running PingFederate as a Service”](#) on page 22).
3. Remove symbolic links:

```
rm /etc/rc3.d/S84<pf_user>
rm /etc/rc4.d/S84<pf_user>
rm /etc/rc5.d/S84<pf_user>
rm /etc/rc0.d/K15<pf_user>
rm /etc/rc1.d/K15<pf_user>
rm /etc/rc2.d/K15<pf_user>
rm /etc/rc6.d/K15<pf_user>
```
4. (Optional) Delete the script used to start and stop the service (see [“Running PingFederate as a Service”](#) on page 22).

Console Navigation

The PingFederate administrator's user interface, the *administrative console*, is built around a system of wizard-like control screens, which are accessed from a top-level portal, the Main Menu.

This chapter covers:

- [“Using the Main Menu”](#) on page 27
- [“Navigating the Administrative Console”](#) on page 30



Note: This information is presented from the viewpoint of an administrative user with full permissions to configure local server settings and partner connections (see [“Account Management”](#) in the “System Administration” chapter of the *PingFederate Administrator's Manual*).

Using the Main Menu

When you log on to PingFederate, you reach the Main Menu, from which you can modify your local server settings or access configuration screens to set up or modify connections with partners (see Figure 6 on page 28).

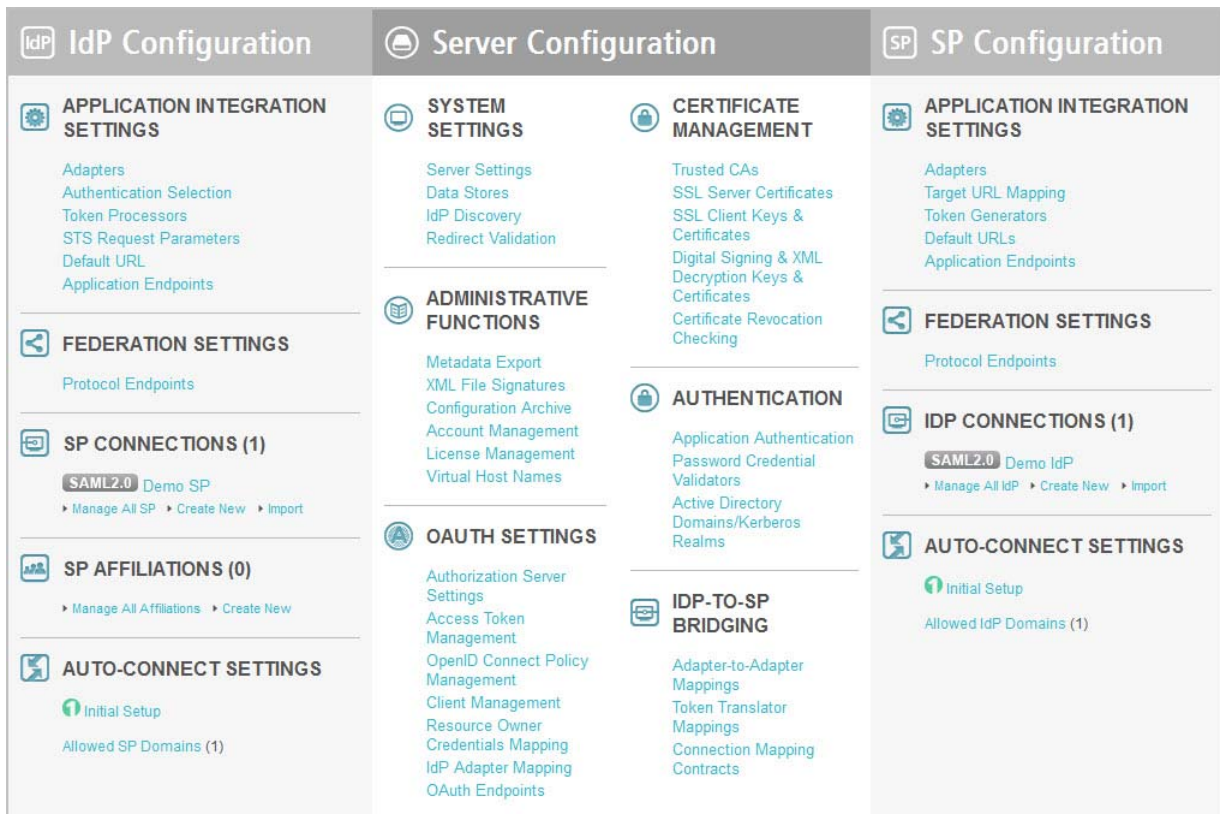


Figure 6: Main Menu (Example)

Note that Main Menu selections depend on your federation role (IdP, SP, or both) and which protocol(s) you are using (see “Choosing Roles and Protocols” in the “System Settings” chapter of the *PingFederate Administrator’s Manual*). Selections also depend on your system permissions (see “Account Management” in the “System Administration” chapter of the *PingFederate Administrator’s Manual*).

Depending on your permissions, you can use the Main Menu to:

- Modify or add to system settings after installation—see the “System Settings” chapter in the *PingFederate Administrator’s Manual*
- Handle system administration functions—see the “System Administration” chapter in the *PingFederate Administrator’s Manual*
- Configure settings for using PingFederate as an OAuth Authorization Server (see the “OAuth Configuration” chapter in the *PingFederate Administrator’s Manual*)
- Manage security certificates, authentication for applications, and protocol authentication validation—see the “Security Management” chapter in the *PingFederate Administrator’s Manual*
- Configure connections and other IdP or SP settings—see the “Identity Provider SSO Configuration” or the “Service Provider SSO Configuration” chapters, respectively, in the *PingFederate Administrator’s Manual*

Navigating the Administrative Console

PingFederate’s configuration screens are designed to guide you through the process of setting up and maintaining your server. This configuration design provides three major benefits:

First, given the complicated security considerations and elaborate requirements under the SAML specifications, setting up an identity federation is complex. The PingFederate setup screens provide a step-by-step mechanism that minimizes the chance of overlooking critical settings.

Second, setting up a federation involves many choices based on your agreement with your partner (see “[Federation Planning Checklist](#)” in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*). PingFederate presents these choices in an organized way and then takes you along the right path, presenting only the steps you need to take based on previous choices.

Finally, like most complex network configurations, federation setup involves many interdependencies. PingFederate keeps track of these for you: when you make a change, the system finds related changes and takes you to the relevant screens.



Caution: Do not use the browser’s Back, Refresh, or Forward buttons. Instead, use the navigation buttons in the lower right portion of the configuration screens (see “[Console Buttons](#)” on page 31).

About Tasks and Steps

Each broad configuration area is broken down into a series of tasks. Each task consists of a sequence of steps. The tasks and steps appear in the top portion of the screen, as shown in Figure 7.

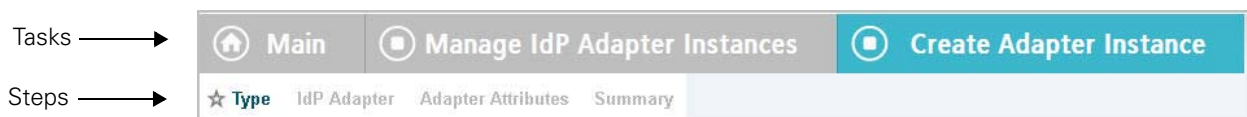


Figure 7: Tasks and Steps (Example)

Notice that steps you have not yet reached are grayed out. After you complete a step, you can click it to go back. When all the steps are completed, you can click any of them to review your work or make changes.



Important: Be sure to click **Save** (if available) when you reach the last step of a task (if you want to save changes), or if you have finished editing a step.

As you traverse steps for each task, you will notice that some provide buttons that branch to dependent, multi-step tasks. In addition, on some screens buttons provide shortcuts to supporting tasks, typically those used for global settings—for

example, as you set up a connection to a partner, you might need to import a certificate into your trusted store (see “[Trusted Certificate Authorities](#)” in the “Security Management” chapter of the *PingFederate Administrator’s Manual*).

In either case, when you change tasks, the transitional step or related global task appears as the current task, and the steps change accordingly.



Caution: Clicking **Cancel** on any screen discards all new unsaved entries or changes for all steps shown for the current task and returns you to the screen from which you accessed the task.

Console Buttons

The navigational and control buttons at the bottom of the administrative console screen change depending on where you are in the configuration process. The following table describes the behavior of these buttons.

Table 2: Administrative Console Buttons

Button	Description
Save	Stores information for all steps completed for the current task or any changes made for the current step; returns to the screen from the which the task or step was accessed (see “ About Tasks and Steps ” on page 30). This button is available only when the Save operation is valid within the current context.
Done	Marks as complete all steps for a current task, but does not save the configuration (because further tasks or steps are necessary); to save entries or changes, click Save (or continue the configuration until you see a Save button). When creating a new connection, click Save Draft (see below).
Save Draft	Stores a new connection configuration for all steps completed up to the current screen in the configuration flow. To return to the draft, click Manage All IdP under SP Connections (or Manage All SP under IdP Connections) on the Main Menu and then select the draft from the connection list.
Cancel	Returns to the screen from which the current task was accessed; discards any information newly entered or modified for all steps in the task (see “ About Tasks and Steps ” on page 30).
Previous	Returns to the previous step (when applicable).
Next	Moves display forward to the next step (when applicable), if all required information is complete in the current step.

Supported Standards

PingFederate provides flexible, integrated support for all versions of the Security Assertion Markup Language (SAML) protocol, from 1.0 through 2.0, OAuth, and for WS-Trust, which underlies the PingFederate [STS](#) for Web Services. In addition, PingFederate supports the WS-Federation browser-based, “passive” protocol using SAML assertions as SSO-enabling security tokens.

This chapter describes:

- [“Federation Roles”](#) on page 33
- [“Terminology”](#) on page 34
- [“SAML 1.x Profiles”](#) on page 36
- [“SAML 2.0 Profiles”](#) on page 39
- [“WS-Federation”](#) on page 51
- [“About Account Linking”](#) on page 53
- [“Web Services Standards”](#) on page 54
- [“OAuth 2.0”](#) on page 57
- [“System for Cross-domain Identity Management \(SCIM\)”](#) on page 60
- [“Transport and Message Security”](#) on page 60

Federation Roles

The most recent sets of standards, SAML 2.0 and WS-Federation, define two roles in an identity federation partnership: an Identity Provider (IdP) and a Service Provider (SP).



Note: Earlier SAML 1.x specifications used the terms Asserting Party (for IdP) and Relying Party (for SP). For consistency and clarity, however, PingFederate adopts the later terms IdP and SP across all specifications.

A third role, defined in the specifications and available in PingFederate, is that of an IdP Discovery provider.

Identity Provider

An IdP, also called the “SAML authority,” is a system entity that authenticates a user, or “SAML subject,” and transmits referential identity information based on that authentication.



Note: The SAML subject may be a person, a Web application, or a Web server. Since the subject is often a person, the term “user” is generally employed throughout this manual.

Service Provider

An SP is the consumer of identity information provided by the IdP. Based on trust, technical agreements, and verification of adherence to protocols, SP applications and systems determine whether (or how) to use information contained in a SAML assertion.

IdP Discovery Provider

This role provides an IdP look-up service that can be incorporated into the implementation of either an IdP or an SP, or it can be employed as a stand-alone server (see “[IdP Discovery](#)” on page 51).

Terminology

The SAML specifications provide a system of building blocks and support components for achieving secure data exchange in an identity federation. These include:

- [Assertions](#)
- [Bindings](#)
- [Profiles](#)
- [Metadata](#)
- [Authentication Context](#)

Assertions

Assertions are XML documents sent from an IdP to an SP. Each assertion contains identifying information about a user who has initiated an SSO request.

Bindings

A SAML binding describes the way messages are exchanged using transport protocols. PingFederate supports the following bindings:

- **HTTP POST** – Describes how SAML messages are transported in HTML form-control content, which uses a base-64 format.
- **HTTP Artifact** – Describes how to use an [artifact](#) to represent a SAML message. The artifact can be transported via an HTML form control or a query string in the URL.

- **HTTP Redirect (SAML 2.0)** – Describes how SAML messages are transported using HTTP 302 status-code response messages.
- **SOAP (SAML 2.0)** – Describes how SAML messages are to be transferred across the back channel (Simple Object Access Protocol).

Profiles

Profiles describe processes and message flows combining assertions, request/response message specifications, and bindings to achieve a specific desired functionality or use case. Because profiles define the application of the specifications and therefore play a large part in PingFederate, most of the rest of this chapter is devoted to them, starting with [“SAML 1.x Profiles”](#) on page 36.

Metadata

SAML 2.0 defines an XML schema to standardize metadata to facilitate the exchange of configuration information among federation partners. This information includes, for example, profile and binding support, connection endpoints, and certificate information. (See [“Exporting Metadata”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*.)

Whether you are exporting or importing a metadata file, PingFederate supports the use of XML digital signatures to ensure the integrity of the data (see [“Signing XML Files”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*).

Authentication Context

Before allowing access to a protected resource, an SP may want information surrounding how the user was originally authenticated by the IdP, in addition to the assertion itself. The SP may use this information for an access control decision or to provide an audit trail for regulatory or security-policy compliance.

The SAML 2.0 specification provides an XML schema whereby partners can create authentication-context declarations. Partners may choose to reference a URI to implement a set of classes provided by the specification to help categorize and simplify context interpretation (see the OASIS document: [saml-authn-context-2.0-os.pdf](#)). However, it is up to partners to decide if additional authentication context is required and if these classes supply an adequate description. For SAML 1.x, the authentication context (called “AuthenticationMethod”), if used, must be specified as a URI (see, for example, [oasis-sssc-saml-core-1.1.pdf](#)).

An administrator can configure PingFederate, acting as an IdP, to include authentication context in assertions. For information about this configuration, see [“Creating an Attribute Contract”](#) in the “Identity Provider SSO Configuration” chapter of the *PingFederate Administrator’s Manual* or [“Defining an STS Attribute Contract”](#) in the “WS-Trust STS Configuration” chapter.

Alternatively, several PingFederate integration kits provide methods that can be used by the developer to insert authentication context from external IdP applications into the assertion (see [“SSO Integration Kits and Adapters”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*). Conversely, the SP developer can call methods for extracting authentication context from an assertion. It is up to the SP developer and application to create access control or other processing based on the context.

Check the *User Guide* for your integration kit to see if this feature is supported.

For more information on configuring authentication context for an adapter instance, see [“Selecting an Authentication Context”](#) in the “Identity Provider SSO Configuration” chapter of the *PingFederate Administrator’s Manual*.

Browser-based SSO

Browser-based SSO includes SAML 1.x, 2.0, and WS-Federation and provides standards-based SSO, Single Logout, Attribute Query and XASP, and the WS-Federation Passive Requestor Profile for SP-initiated SSO.

SAML 1.x Profiles

SAML 1.0 and 1.1 profiles provide for browser-based SSO, initiated by an IdP, using either the POST or artifact bindings.

In addition, the specifications provide for a non-normative SP-initiated scenario (called “destination-first”), which allows Web developers to create applications that enable a user to initiate SSO from the SP site.

SSO--Browser-Post

In this scenario, a user is logged on to the IdP and attempts to access a resource on a remote SP server. The SAML assertion is transported to the SP via HTTP POST.

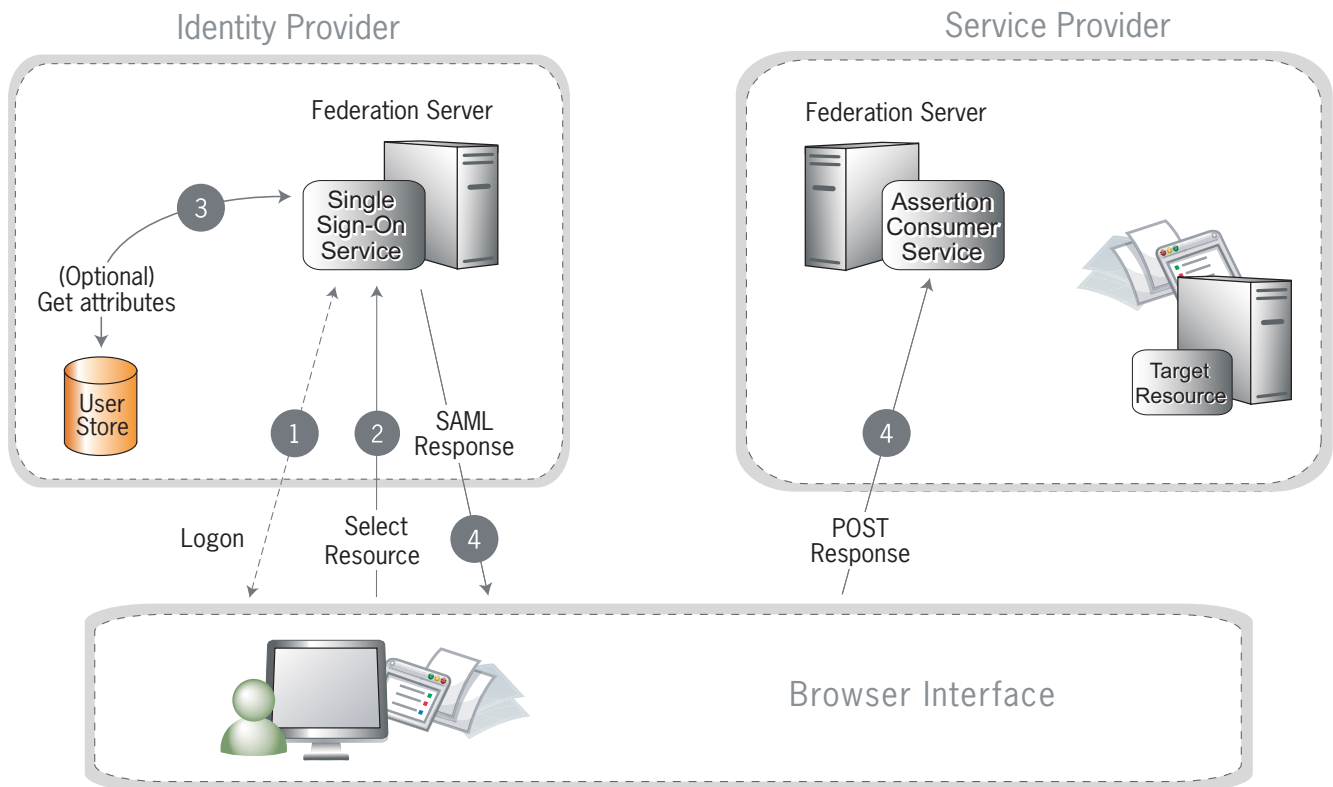


Figure 8: Browser/POST Profile

Processing Steps:

1. A user has logged on to the IdP.
(If a user has not yet logged on for some reason, he or she is challenged to do so at step 2).
2. The user clicks a link or otherwise requests access to a protected SP resource.
3. Optionally, the IdP retrieves attributes from the user data source.
4. The IdP's SSO service returns an HTML form to the browser with a SAML response containing the authentication assertion and any additional attributes. The browser automatically posts the HTML form back to the SP.



Note: SAML specifications require that POST responses be digitally signed.

5. (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

SSO--Browser-Artifact

In this scenario, the IdP sends a SAML artifact to the SP via either HTTP POST or a redirect (shown in diagram). The SP uses the artifact to obtain the associated SAML response from the IdP.

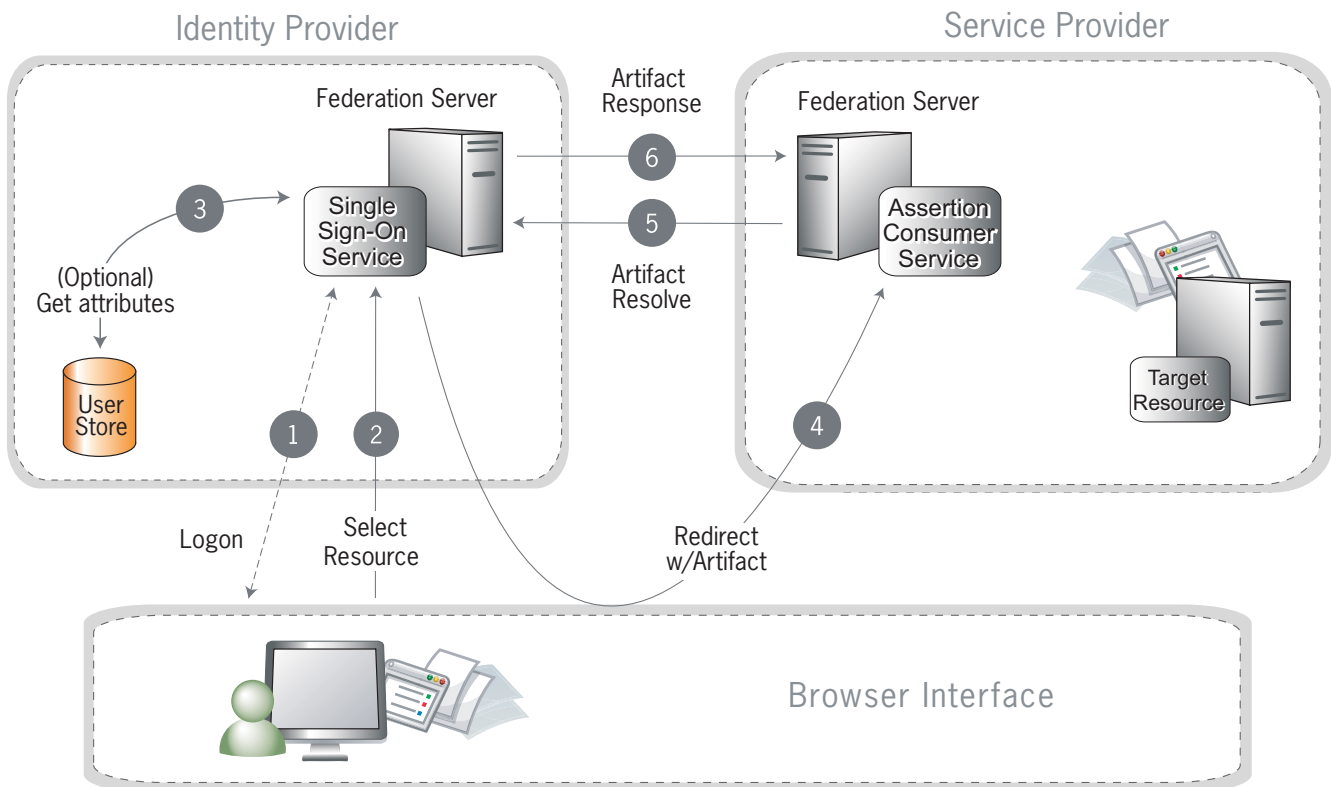


Figure 9: SSO: Browser/Artifact Profile

Processing Steps:

1. A user has logged on to the IdP.
(If a user has not yet logged on for some reason, he or she is challenged to do so at step 2).
2. The user clicks a link or otherwise requests access to a protected SP resource.
3. Optionally, the IdP retrieves attributes from the user data store.
4. The IdP federation server generates an assertion, creates an artifact, and sends an HTTP redirect containing the artifact through the browser to the SP's Assertion Consumer Service (ACS).
5. The ACS extracts the Source ID from the SAML artifact and sends an artifact-resolve message to the identity federation server's Artifact Resolution Service (ARS).
6. The ARS sends a SAML artifact response message containing the previously generated assertion.
7. (Not shown) If a valid assertion is received, the SP establishes a session and redirects the browser to the target resource.

SP-Initiated ("Destination-First") SSO

In an SP-initiated (a.k.a. "destination-first") transaction the user is connected to an SP site and attempts to access a protected resource in the SP domain. The user might have an account at the SP site but according to federation agreement, authentication is managed by the IdP. The SP sends an authentication request to the IdP.

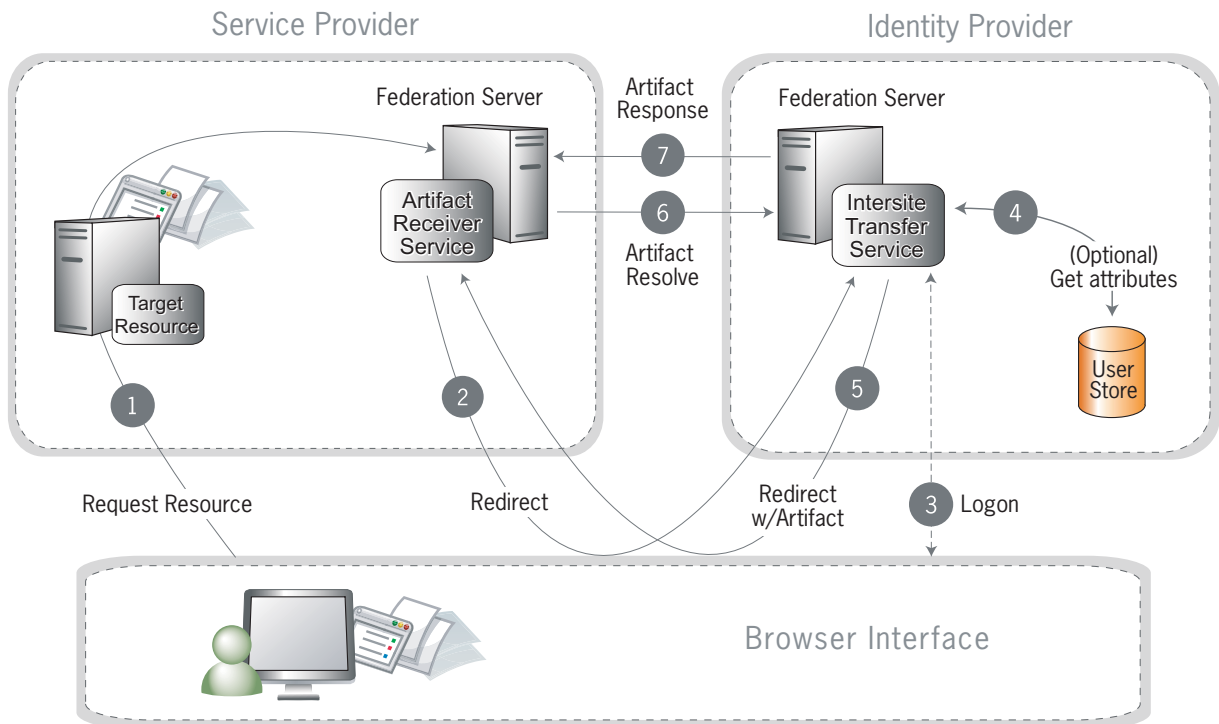


Figure 10: SP-Initiated SSO

Processing Steps:

1. The user requests access to a protected SP resource. The request is redirected to the federation server (e.g., PingFederate) to handle authentication.
2. The federation server sends a SAML request for authentication to the IdP's SSO service (also called the Intersite Transfer Service).
3. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.)
5. The IdP's Intersite Transfer Service returns an [artifact](#), representing the SAML response, to the SP.
6. The SP's artifact handling service sends a SOAP request with the artifact to the IdP's artifact resolver endpoint.
7. The IdP resolves the artifact and returns the corresponding SAML response with the SSO assertion.
8. (Not shown) If the assertion is valid, the SP establishes a session for the user and redirects the browser to the target resource.

SAML 2.0 Profiles

PingFederate supports these major profiles defined under the SAML 2.0 standard:

- [Single Sign-on](#)
- [Single Logout](#)
- [Attribute Query and XASP](#)
- [IdP Discovery](#)

Single Sign-on

SAML 2.0 substantially increases the number of possible SSO profile variations by fully enabling SP-initiated transactions. When SP- and IdP-initiated protocols are paired with transport [binding](#) specifications, the combinations result in eight practical SSO scenarios:

- [SP-Initiated SSO--POST-POST](#)
- [SP-Initiated SSO--Redirect-POST](#)
- [SP-Initiated SSO--Artifact-POST](#)
- [SP-Initiated SSO--POST-Artifact](#)
- [SP-Initiated SSO--Redirect-Artifact](#)
- [SP-Initiated SSO--Artifact-Artifact](#)
- [IdP-Initiated SSO--POST](#)
- [IdP-Initiated SSO--Artifact](#)

SP-Initiated SSO--POST-POST

In this scenario a user attempts to access a protected resource directly on an SP Web site without being logged on. The user does not have an account on the SP site, but does have a federated account managed by a third-party IdP. The SP sends an authentication request to the IdP. Both the request and the returned SAML assertion are sent through the user's browser via HTTP POST.

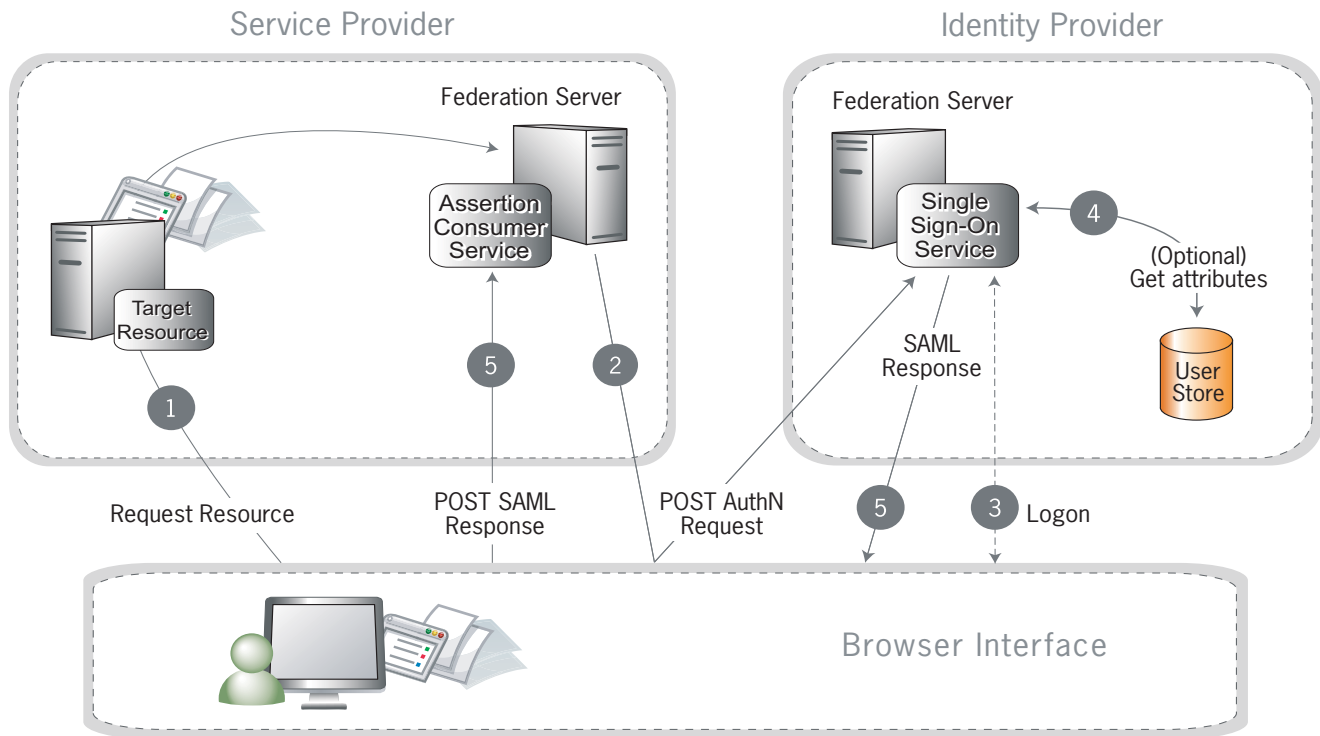


Figure 11: SP-Initiated SSO: POST/POST

Processing Steps:

1. The user requests access to a protected SP resource. The request is redirected to the federation server to handle authentication.
2. The federation server sends an HTML form back to the browser with a SAML request for authentication from the IdP. The HTML form is automatically posted to the IdP's SSO service.
3. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.)

- The IdP's SSO service returns an HTML form to the browser with a SAML response containing the authentication assertion and any additional attributes. The browser automatically posts the HTML form back to the SP.



Note: SAML specifications require that POST responses be digitally signed.

- (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

SP-Initiated SSO--Redirect-POST

In this scenario, the SP sends an HTTP redirect message to the IdP containing an authentication request. The IdP returns a SAML response with an assertion to the SP via HTTP POST.

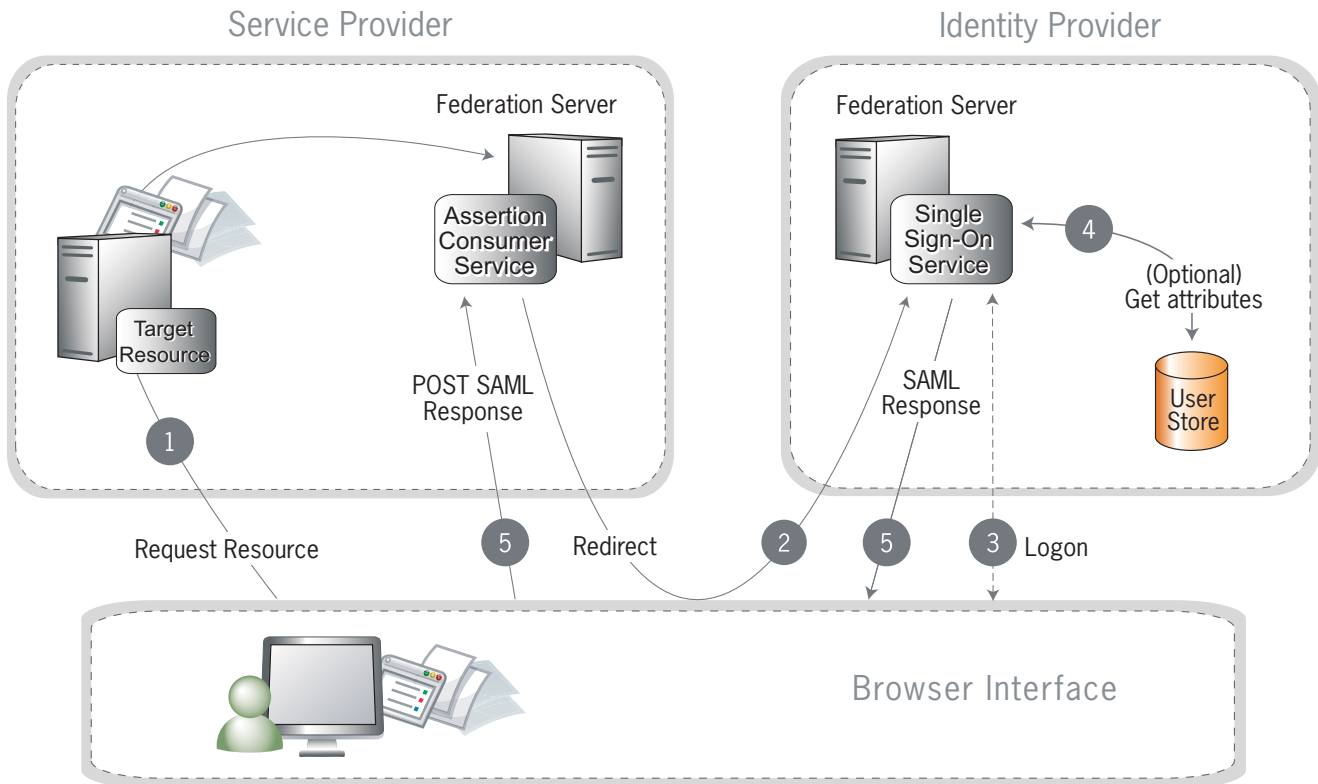


Figure 12: SP-Initiated SSO: Redirect/POST

Processing Steps:

- A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.
- The SP returns an HTTP redirect (code 302 or 303) containing a SAML request for authentication through the user's browser to the IdP's SSO service.
- If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.

4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see “About Attributes” in the “Key Concepts” chapter of the PingFederate Administrator’s Manual.)
5. The IdP’s SSO service returns an HTML form to the browser with a SAML response containing the authentication assertion and any additional attributes. The browser automatically posts the HTML form back to the SP.



Note: SAML specifications require that POST responses be digitally signed.

6. (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

SP-Initiated SSO--Artifact-POST

In this scenario, the SP sends a SAML artifact to the IdP via an HTTP redirect. The IdP uses the artifact to obtain an authentication request from the SP’s SAML artifact resolution service. The IdP returns a SAML response to the SP via HTTP POST.

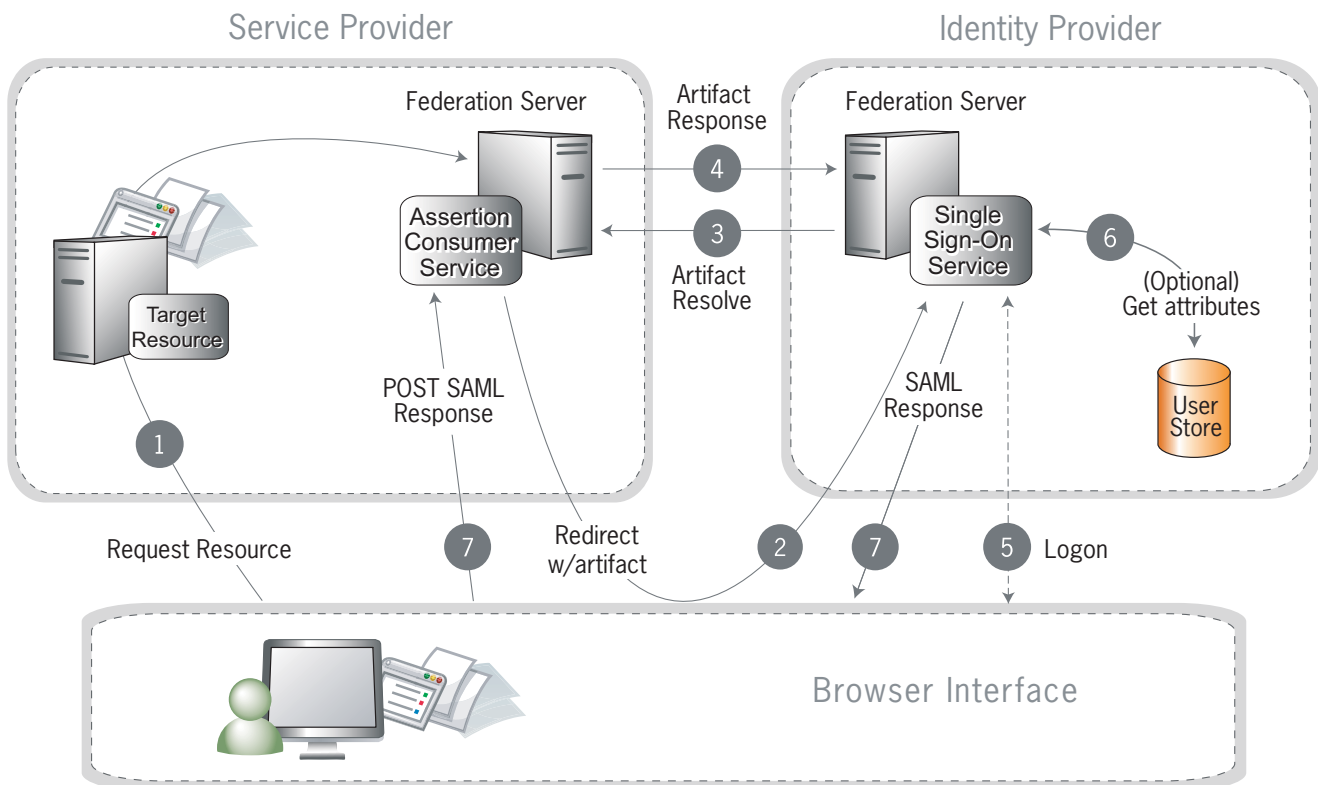


Figure 13: SP-Initiated SSO: Artifact/POST

Processing Steps:

1. A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.

2. The SP generates an authentication request and creates an artifact. The SP sends an HTTP redirect containing the artifact through the user's browser to the IdP's SSO service.



Note: The artifact contains the source ID of the SP's artifact resolution service and a reference to the authentication.

3. The SSO service extracts a source ID from the SAML artifact and sends a SAML artifact-resolve message over [SOAP](#) containing the artifact to the SP's [Artifact Resolution Service \(ARS\)](#).



Note: The SP and IdP's source IDs and remote artifact resolution services are mapped according to the federation agreement made prior to this action.

4. The SP's ARS returns a SAML message containing the previously generated authentication request.
5. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
6. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the *PingFederate Administrator's Manual*.)
7. The IdP's SSO service returns an HTML form to the browser with a SAML response containing the authentication assertion and any additional attributes. The browser automatically posts the HTML form back to the SP.



Note: SAML specifications require that POST responses be digitally signed.

8. (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

SP-Initiated SSO--POST-Artifact

In this scenario, the SP sends an authentication request to the IdP via HTTP POST. The returned SAML assertion is redirected through the user's browser. The response contains a SAML [artifact](#).

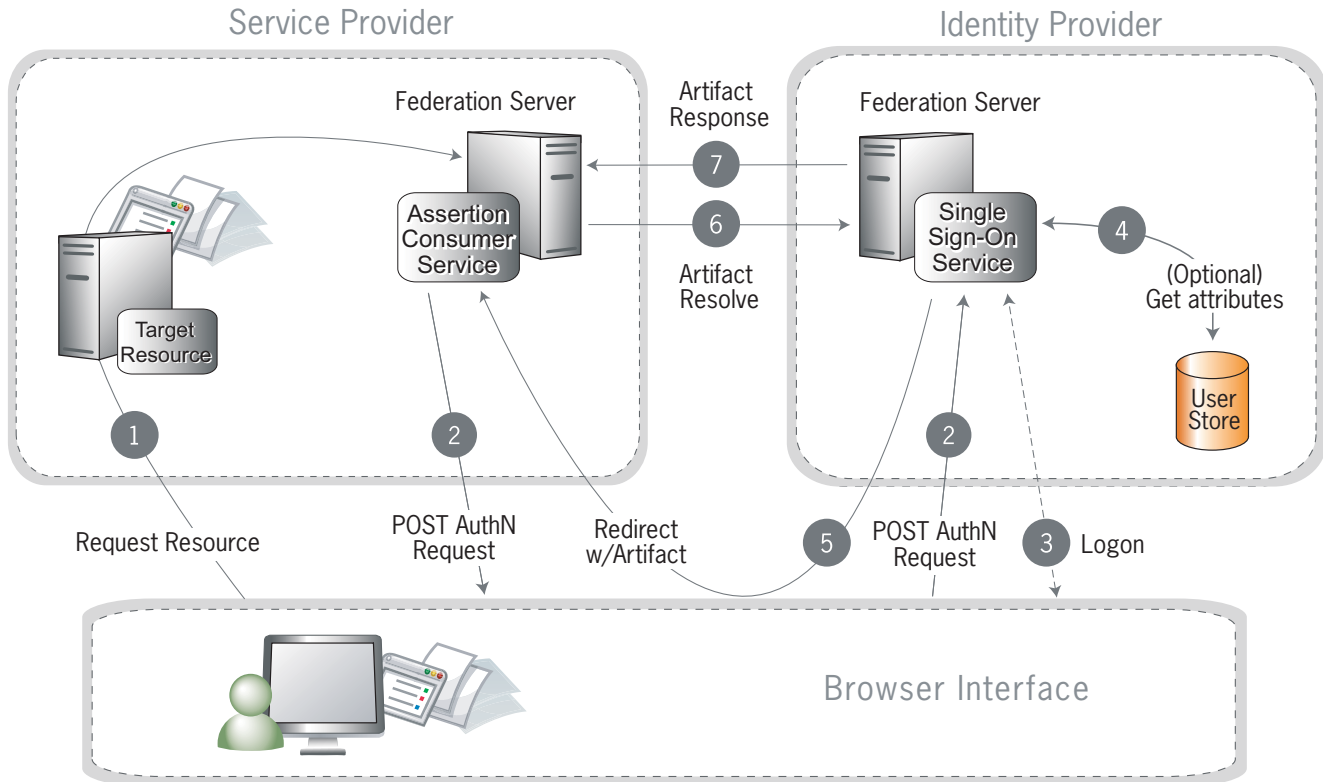


Figure 14: SP-Initiated SSO: POST/Artifact

Processing Steps:

1. A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.
2. The federation server sends an HTML form back to the browser with a SAML request for authentication from the IdP. The HTML form is automatically posted to the IdP's SSO service.
3. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.)
5. The IdP federation server generates an assertion, creates an artifact, and sends an HTTP redirect containing the artifact through the browser to the SP’s [Assertion Consumer Service \(ACS\)](#).

6. The ACS extracts the source ID from the SAML artifact and sends an artifact-resolve message to the federation server's [Artifact Resolution Service](#) (ARS).
7. The ARS sends a SAML artifact response message containing the previously generated assertion.
8. (Not shown) If a valid assertion is received, a session is established on the SP and the browser is redirected to the target resource.

SP-Initiated SSO--Redirect-Artifact

In this scenario, the SP sends an HTTP redirect message to the IdP containing a request for authentication. The IdP returns an [artifact](#) via HTTP redirect. The SP uses the artifact to obtain the SAML response.

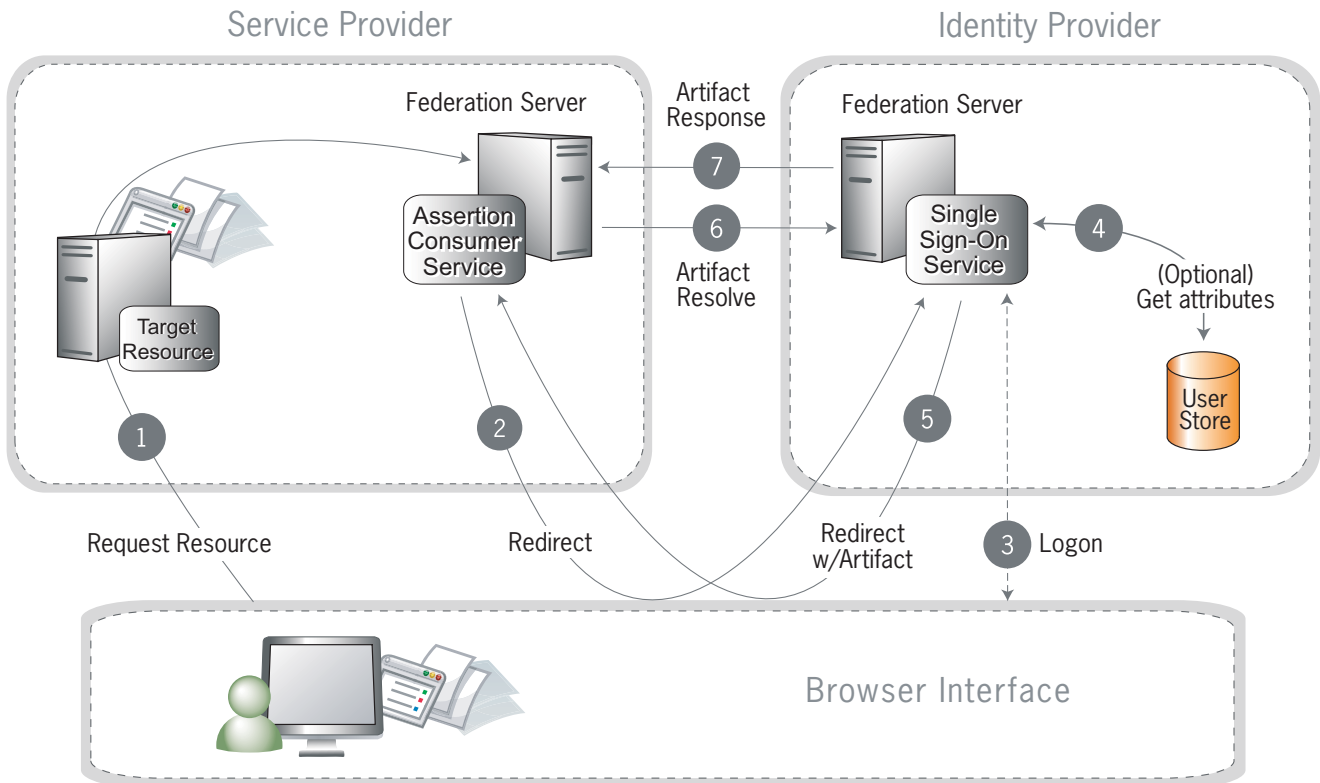


Figure 15: SP-Initiated SSO: Redirect/Artifact

Processing Steps:

1. A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.
2. The SP returns an HTTP redirect (code 302 or 303) containing a SAML request for authentication through the user's browser to the IdP's SSO service.
3. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About](#)

Attributes” in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*.)

5. The IdP federation server generates an assertion, creates an artifact, and sends an HTTP redirect containing the artifact through the browser to the SP’s Assertion Consumer Service (ACS).
6. The ACS extracts the Source ID from the SAML artifact and sends an artifact-resolve message to the identity federation server’s Artifact Resolution Service (ARS).
7. The ARS sends a SAML artifact response message containing the previously generated assertion.
8. (Not shown) If a valid assertion is received, the SP establishes a session and redirects the browser to the target resource.

SP-Initiated SSO--Artifact-Artifact

In this scenario, the SP sends a SAML *artifact* to the IdP via an HTTP redirect. The IdP uses the artifact to obtain an authentication request from the SP. Then the IdP sends another artifact to the SP, which the SP uses to obtain the SAML response.

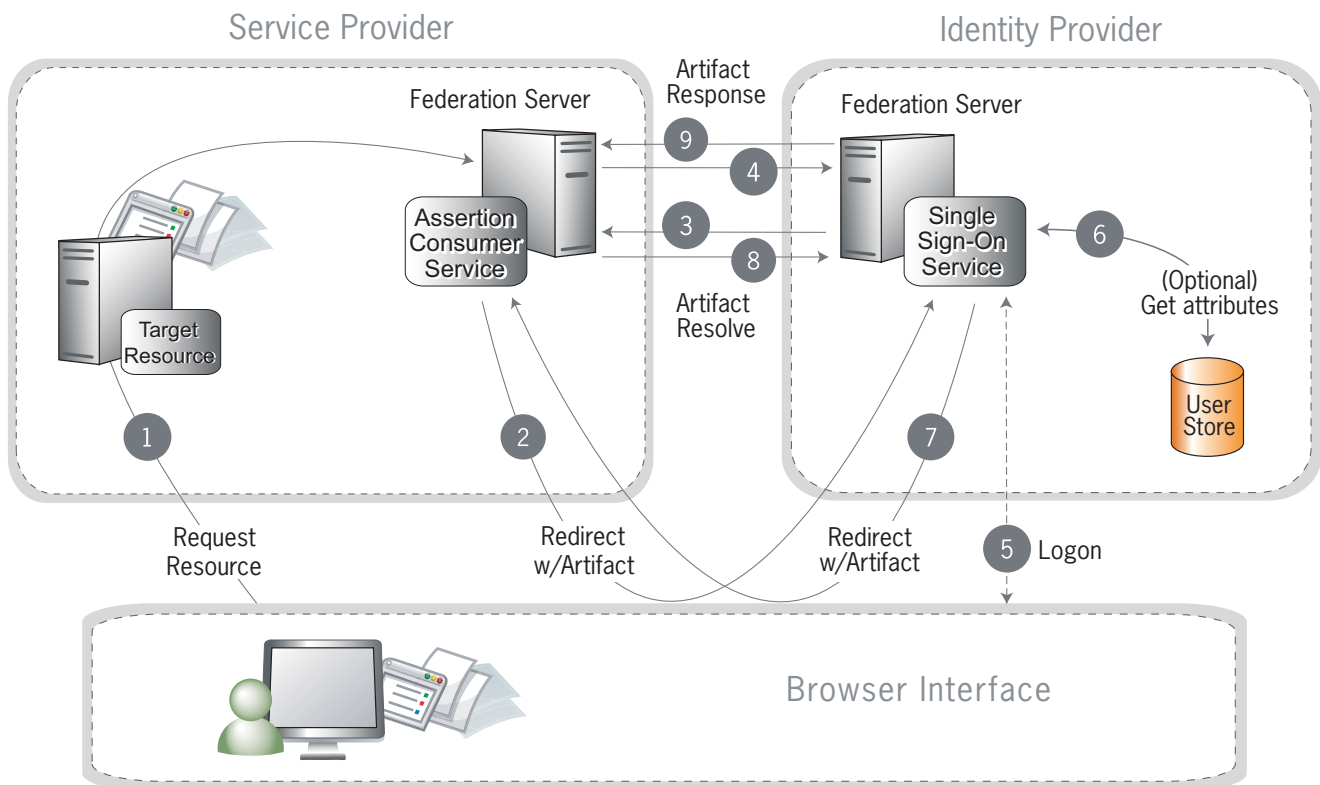


Figure 16: SP-Initiated SSO: Artifact/Artifact

Processing Steps:

1. A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.

2. The ACS generates an authentication request and creates an artifact. It sends an HTTP redirect containing the artifact through the user's browser to the IdP's SSO service.



Note: The artifact contains the source ID of the SP's artifact resolution service and a reference to the authentication request.

3. The SSO service extracts the source ID from the SAML artifact and sends a SAML artifact resolve message containing the artifact to the SP's artifact resolution service.



Note: The SP and IdP's source IDs and remote artifact resolution services are mapped according to the federation agreement prior to this action.

4. The SP's artifact resolution service sends back a SAML artifact response message containing the previously generated authentication request.
5. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
6. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the *PingFederate Administrator's Manual*.)
7. The IdP federation server generates an assertion, creates an artifact, and sends an HTTP redirect containing the artifact through the browser to the SP's Assertion Consumer Service (ACS).
8. The ACS extracts the Source ID from the SAML artifact and sends an artifact-resolve message to the identity federation server's Artifact Resolution Service (ARS).
9. The ARS sends a SAML artifact response message containing the previously generated assertion.
10. (Not shown) If a valid assertion is received, the SP establishes a session and redirects the browser to the target resource.

IdP-Initiated SSO--POST

In this scenario, a user is logged on to the IdP and attempts to access a resource on a remote SP server. The SAML assertion is transported to the SP via HTTP POST.

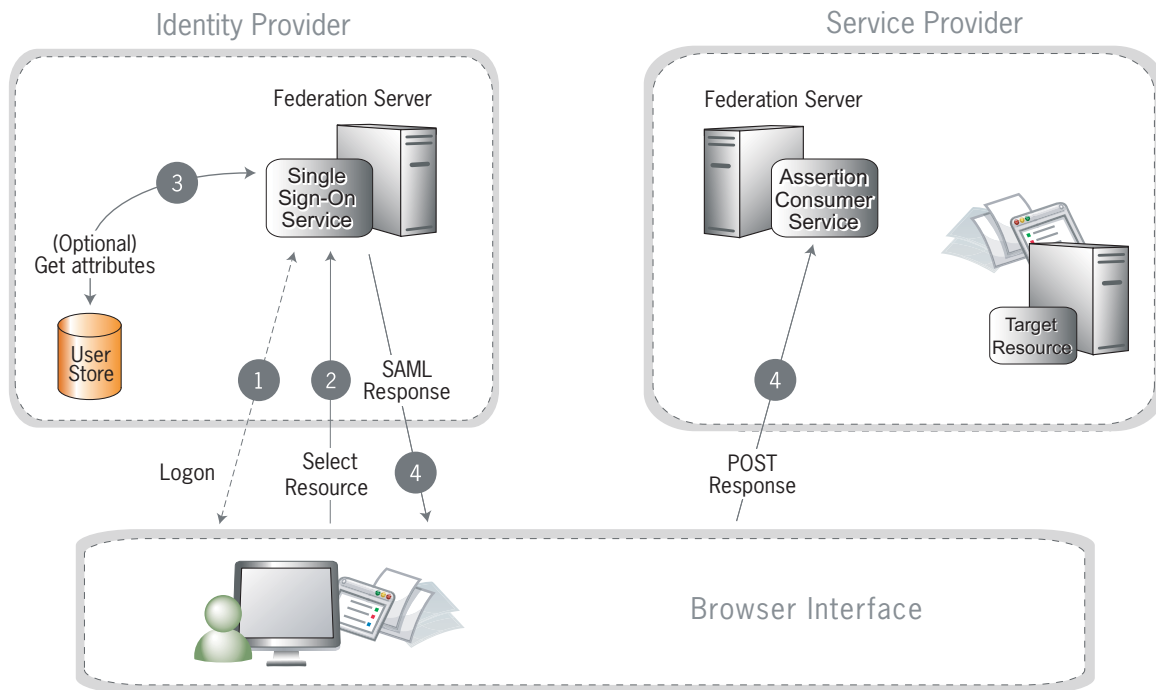


Figure 17: IdP-Initiated SSO: POST

Processing Steps:

1. A user has logged on to the IdP.
(If a user has not yet logged on for some reason, he or she is challenged to do so at step 2).
2. The user clicks a link or otherwise requests access to a protected SP resource.
3. Optionally, the IdP retrieves attributes from the user data store.
4. The IdP's SSO service returns an HTML form to the browser with a SAML response containing the authentication assertion and any additional attributes. The browser automatically posts the HTML form back to the SP.



Note: SAML specifications require that POST responses be digitally signed.

5. (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

IdP-Initiated SSO--Artifact

In this scenario, the IdP sends a SAML artifact to the SP via an HTTP redirect. The SP uses the artifact to obtain the associated SAML response from the IdP.

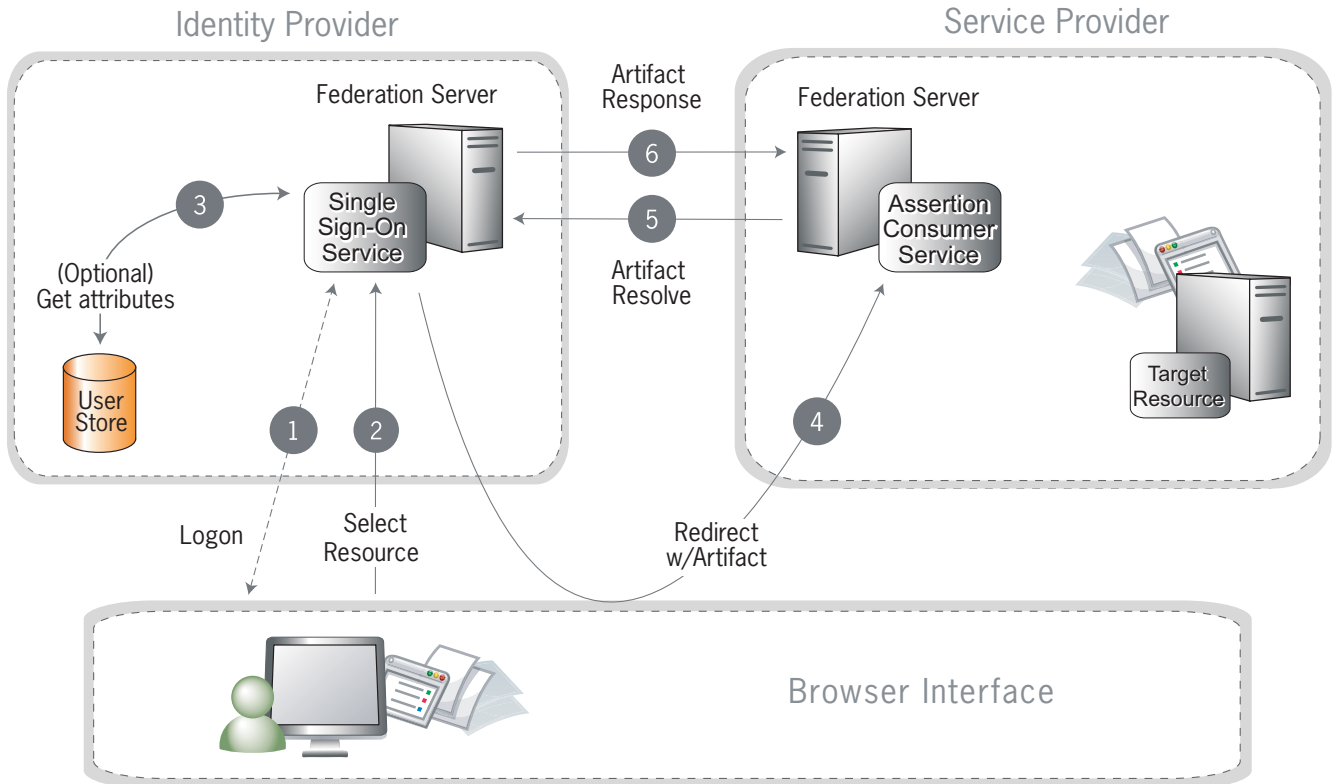


Figure 18: IdP-Initiated SSO: Artifact

Processing Steps:

1. A user has logged on to the IdP.
(If a user has not yet logged on for some reason, he or she is challenged to do so at step 2).
2. The user clicks a link or otherwise requests access to a protected SP resource.
3. Optionally, the IdP retrieves attributes from the user data store.
4. The IdP federation server generates an assertion, creates an artifact, and sends an HTTP redirect containing the artifact through the browser to the SP's Assertion Consumer Service (ACS).
5. The ACS extracts the Source ID from the SAML artifact and sends an artifact-resolve message to the identity federation server's Artifact Resolution Service (ARS).
6. The ARS sends a SAML artifact response message containing the previously generated assertion.
7. (Not shown) If a valid assertion is received, the SP establishes a session and redirects the browser to the target resource.

Single Logout

The single logout (SLO) profile enables a user to log out of all participating sites in a federated session nearly simultaneously. The user may log out globally from any site, whether SP or IdP, as determined by respective Web applications. The associated IdP federation deployment handles all logout requests and responses for participating sites.

The logout messages may be transported using any combination of [bindings](#) described for SSO (POST, artifact, or redirect). Refer to the diagrams under “[Single Sign-on](#)” on page 39 for illustrations of these message flows.

About Session Clean-up

When an SP receives an SLO request from an IdP, the session creation [adapter\(s\)](#) you are using must handle any session clean-up with respect to the local application. For more information about adapters, see “[SSO Integration Kits and Adapters](#)” in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.

Attribute Query and XASP

The SAML 2.0 Attribute Query profile allows an SP to request user attributes from an IdP in a secure transaction separate from SSO. The IdP, acting as an *Attribute Authority*, accepts Attribute Queries, performs a data-store lookup into a user repository such as an LDAP directory, provides values to the requested attributes, and generates an Attribute Response back to the originating SP requester. The SP then returns the attributes to the requesting application.



Tip: When privacy is required for sensitive attributes, you can configure PingFederate to obfuscate (mask) their values in the server and transaction logs (see “[Attribute Masking](#)” in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*).

Since Web SSO is distinct from the Attribute Query use case, you can configure PingFederate servers to implement either or both of these profiles without regard to the other.

The X.509 Attribute Sharing Profile (XASP) defines a specialized extension of the general Attribute Query profile. The XASP specification enables organizations with an investment in PKI (Public Key Infrastructure) to issue and receive Attribute Queries based on user-certificate authentication.

Under XASP a user authenticates directly with an SP application by providing his or her X.509 certificate (see “[Authentication](#)” in the “Security Management” chapter of the *PingFederate Administrator’s Manual*). Once the user is authenticated, the SP application requests additional user attributes by contacting the SP PingFederate server. A portion of the user’s X.509 certificate is included in the request and may be used to determine the correct IdP to use as the source of the requested attributes (see “[Attribute Requester Mapping](#)” in the “Service Provider SSO Configuration” chapter of the *PingFederate Administrator’s Manual*). Finally, the SP generates an Attribute Query and transmits it to the IdP over the SOAP back channel.

Because the user arrives at the SP server already authenticated, note that no PingFederate adapter is used in this case (see “[SSO Integration Kits and Adapters](#)” in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*).

IdP Discovery

SAML 2.0 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. This mechanism can be helpful, in particular, in cases where an SP might be a hub for several IdPs in an identity federation.



Tip: In addition to supporting standard IdP Discovery, PingFederate provides a cross-protocol, proprietary mechanism allowing an SP server to write a persistent browser cookie. The cookie contains a reference to the IdP partner with whom the user previously authenticated for SSO. For more information, see [“IdP Discovery Using a Persistent Cookie”](#) in the “System Settings” chapter of the PingFederate *Administrator’s Manual*.

In the standard scenario, when a user requests access to a protected resource on the SP, common-domain browser cookies are used to determine where a user has authenticated in the past. Using this information, a PingFederate server can determine which IdP connection to use for sending an authentication request.

As an IdP Discovery provider, PingFederate can serve in up to three different roles:

- Common domain server
- Common domain cookie writer
- Common domain cookie reader

Each of these roles is necessary to support IdP Discovery. The roles may be distributed across multiple servers at different sites.

Common domain server In this role the PingFederate server hosts a domain that its federation partners share in common. The common domain server allows partners to manipulate browser cookies that exist within that common domain. PingFederate can serve in this role exclusively or as part of either an IdP or an SP federation role, or both.

Common domain cookie writer When PingFederate is acting in an IdP role and authenticates a user, it can write an entry in the common domain cookie, including its federation entity ID. An SP can look up this information on the common domain (not the same location as the common domain server described above).

Common domain cookie reader When PingFederate is acting as an SP and needs to determine the IdPs with whom the user has authenticated in the past, it reads the common domain cookie. Based on the information contained in the cookie, PingFederate can then initiate an SSO authentication request using the correct IdP connection.

WS-Federation

PingFederate supports the WS-Federation Passive Requestor Profile for SP-initiated SSO, enabling interoperability with Microsoft's Active Directory Federation Service (ADFS). This profile provides for straightforward redirects and HTTP GET and

POST methods to transport SAML assertions as security tokens for SSO and logout request and response messages for SLO.



Note: Unlike SAML, WS-Federation consolidates the endpoints for SLO and SSO. So when you set up a WS-Federation connection in PingFederate, both types of transactions are available to an SP Web application that supports them both.

For more information about WS-Federation and the Passive Requestor Profile, see [Web Services Federation Languages](http://specs.xmlsoap.org/ws/2006/12/federation/) (`specs.xmlsoap.org/ws/2006/12/federation/`).

Passive Requestor Profile

This profile permits a user's browser (the passive requestor) to request a security token from an IdP when the user requests access to a protected Web service or other resource at an SP.

Figure 19 illustrates message processing for SSO using WS-Federation.

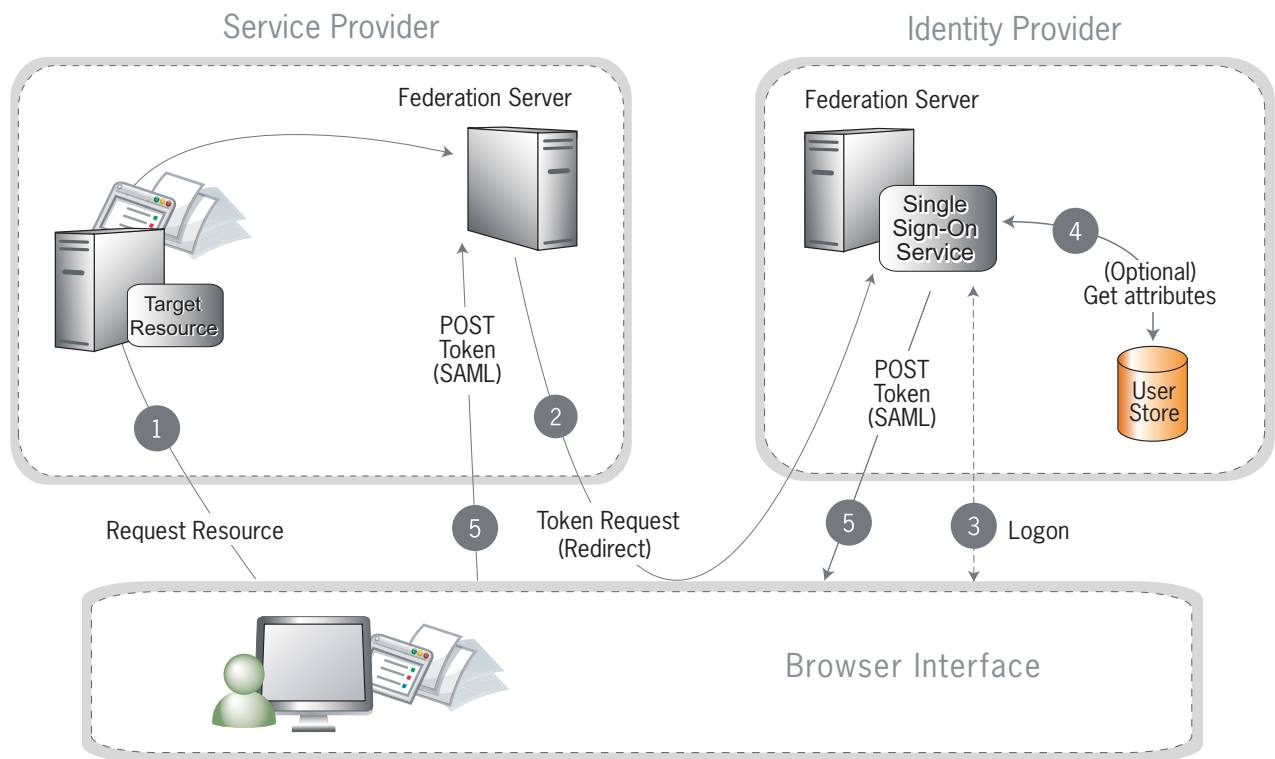


Figure 19: WS-Federation SSO

Processing Steps:

1. A user requests access to a protected SP resource. The user is not logged on to the site. The request is redirected to the federation server to handle authentication.
2. The SP generates a security token request and redirects the browser to the identity provider's WS-Federation implementation.

3. If the user is not already logged on to the IdP site or if re-authentication is required, the IdP asks for credentials (e.g., ID and password) and the user logs on.
4. Additional information about the user may be retrieved from the user data store for inclusion in the SAML response. (These attributes are predetermined as part of the federation agreement between the IdP and the SP—see [“About Attributes”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.)
5. The federation server creates a response containing a signed SAML assertion and returns it to the SP via POST.
6. (Not shown) If the signature and assertion are valid, the SP establishes a session for the user and redirects the browser to the target resource.

Single logout using WS-Federation is handled in much the same way as with SAML (see [“Single Logout”](#) on page 50); however, HTTP GET/POST is always used as the transport mechanism.

About Account Linking

Account linking provides a means for a user to log on to disparate sites with just one authentication, when the user has established accounts and credentials at each site. This method of effectively interconnecting accounts across domains is supported by all protocols.

Account linking involves a *persistent name identifier* associated with accounts at each participating site. The name identifier, which may be an opaque [pseudonym](#), is conveyed in the [assertion](#). Once established locally, the SP can use the account link to look up the user and provide access without re-authentication.

For more information about account linking, see [“Account Linking”](#) in the “Key Concepts” chapter of the *PingFederate Administrator’s Manual*.

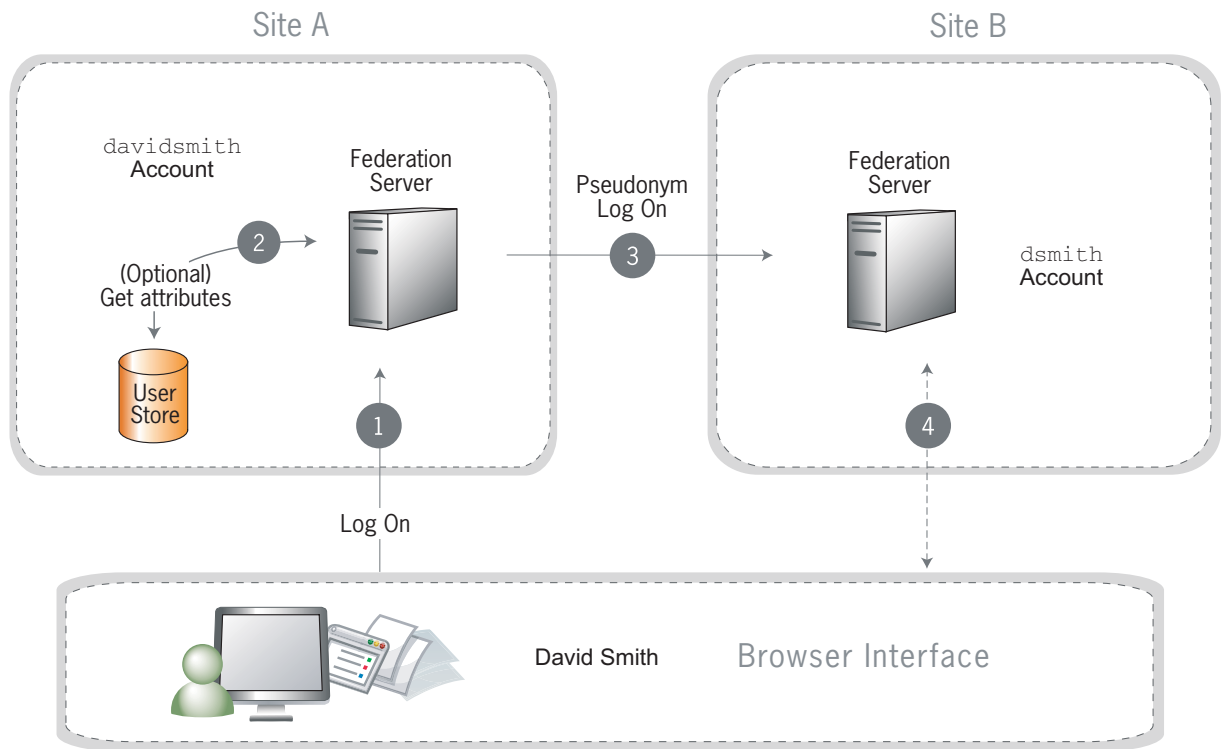


Figure 20: Account Linking

Processing Steps:

1. David Smith logs on to Site A as `daVIDsmith`. He then decides to access his account on Site B via Site A.
2. Optionally, the federation server looks up additional attributes from the data store.
3. The Site A federation server sends a persistent name identifier (possibly a [pseudonym](#)) to Site B, along with any other attributes.
If a pseudonym is used and other attributes are sent, care must be taken not to send attributes that could be used to identify the subject.
4. The federation server on Site B uses the information to associate the pseudonym with the existing account of `dsmith`. (Optionally, David is asked to provide consent to the linking.)
Once the link has been established, it is stored so that David only has to log on to Site A to have access to Site B.

Web Services Standards

The PingFederate WS-Trust STS is designed to interoperate with many different Web Service environments that support varying standards. PingFederate supports multiple versions of SOAP and WS-Trust specifications, and can freely operate with any combinations of these standards simultaneously.

PingFederate supports namespace aliasing to eliminate common trailing-slash inconsistencies for WS-Trust 1.3. (The server does not support namespace aliasing for WS-Trust 2005.)

Supported SOAP/WS-Trust versions and corresponding namespaces are listed in following table:

Table 3 SOAP/WS-Trust Versions

Spec.	Version	Namespace
SOAP	1.1	http://schemas.xmlsoap.org/soap/envelope/
	1.2	http://www.w3.org/2003/05/soap-envelope
WS-Trust	2005	http://schemas.xmlsoap.org/ws/2005/02/trust/
	1.3	http://docs.oasis-open.org/ws-sx/ws-trust/200512/

Web Services Security

Web Services Security (WSS, also WSSE) is a set of specifications defined by the Web Services Security Technical Committee (see www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss) at the OASIS standards organization. WSS defines the XML extensions that can be used to secure Web Service invocations, providing a standard way for partners to add message integrity and confidentiality to their Web Service interactions (see Figure 21). The WSS-defined token profiles describe standard ways of binding security tokens to these messages, enabling a variety of additional capabilities. The WSS technical committee has defined profiles for using SAML assertions, Username, Kerberos, X.509, and other existing security tokens. SSL/TLS is often used in conjunction with deployments of WSS.



Note: The implementation of WSS in the deployment of Web Services identity federations is outside the scope of PingFederate, which provides a standalone, standard means of handling the tokens needed for such federations (see “WS-Trust” below).

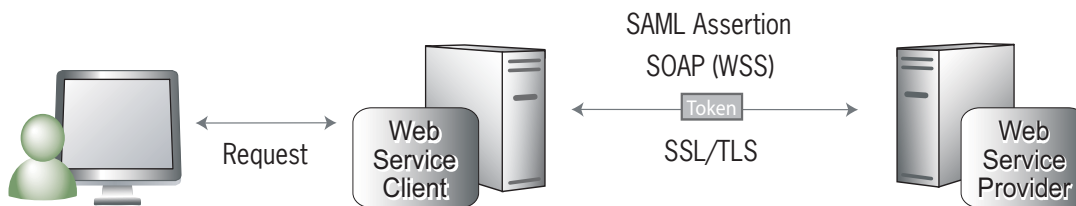


Figure 21: WSS Token Transfer

WS-Trust

WS-Trust comprises a protocol for systems and applications to use when requesting a service to issue, validate, and exchange security tokens. Organizations can leverage this protocol to centralize their security-token processing.

The WS-Trust specification also defines the role of a Security Token Service as the entity responsible for responding to requests using the protocol. In this role, the STS creates new security tokens, validates existing security tokens, and/or exchanges security tokens of one type for those of another (see [Figure 22](#) on page 56).

WS-Trust was created by a consortium of leading platform and security vendors who have contributed the protocol to the OASIS standards organization, where it is managed by the WS-SX (Secure Exchange) technical committee. (See http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx.)

Request Types

The WS-Trust protocol defines two request types that are particularly useful in securing Web Services: “Issue” and “Validate,” often associated with the Web Service Client (WSC) and Web Service Provider (WSP), respectively. The WSC requests that an STS *issue* a SAML token to convey information between the WSC and the WSP. The WSP sends the STS a request to *validate* the incoming token. Optionally, the WSP can request that the STS *issue* a local token for the SP domain.

When issuing and validating security tokens, PingFederate enforces security policies, defined by administrators, generating the token types that are required for a Web Service request to pass between two security domains (whether these domains are within the same organization or in separate organizations).

The following illustration shows an example of a token exchange, using PingFederate to obtain a SAML assertion to be used in the WSS-secured Web Service call.

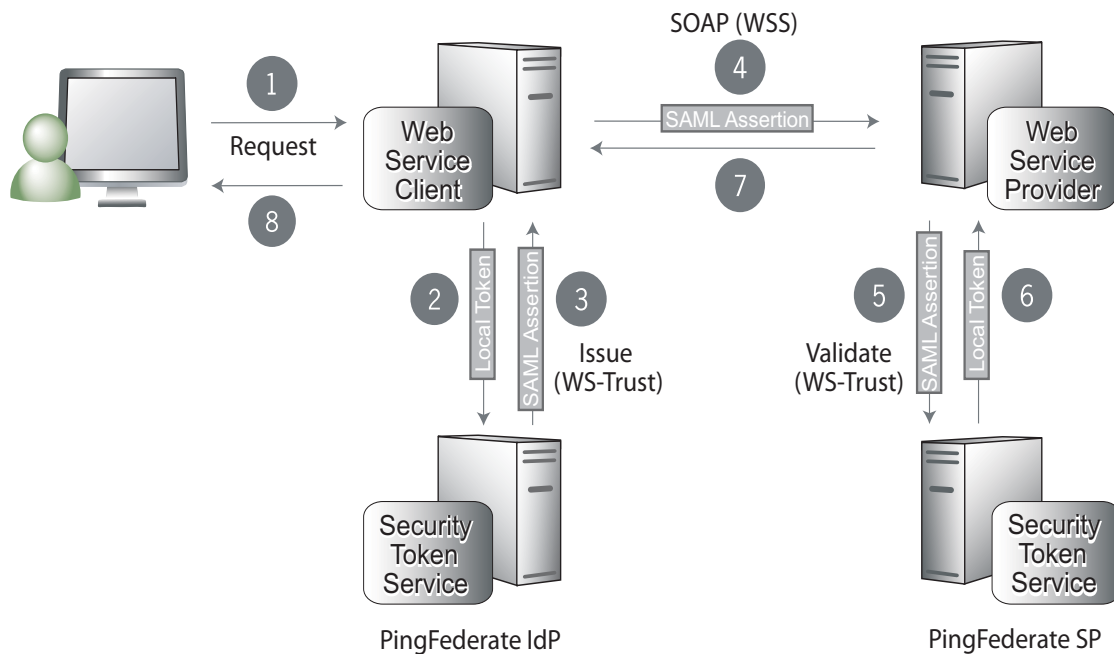


Figure 22: Token Exchange (Example)

Processing Steps:

1. A user requests content from an application.

2. The application acts as a WSC to respond to the user's request. The application calls PingFederate, passing the existing user security token to exchange it for the appropriate SAML [assertion](#).
3. PingFederate verifies the existing security token, creates a new SAML assertion representing the user, and returns it to the requesting application.
4. The application sends a Web Service request to the WSP, including the SAML assertion in a [WSS](#) header.
5. The WSP retrieves the SAML assertion from the WSS header in the incoming request and sends a message to its own deployment of PingFederate to determine if the assertion is valid.
6. PingFederate validates the SAML assertion, creates a new security token for the local domain, and returns the new token to the WSP.
7. The WSP responds to the request according to its policy for the user.
8. The Web application returns an HTML page to the user.



Note: This example shows PingFederate deployed in both the Client and Provider sides of the interaction. However, other deployment options are also supported.

OAuth 2.0

OAuth 2.0 defines a protocol for securing application access to protected resources by issuing access tokens to clients of Representational State Transfer (REST) APIs (and non-REST APIs). Rather than the client directly authenticating to the API using credentials, or the credentials of a user, OAuth enables the client to authenticate by presenting a previously obtained token. The token represents (or contains) a set of attributes and/or policies appropriate to the client and the user. These tokens present less of a security and privacy risk than using secrets (or passwords) directly on the API call. The attributes are used by the API to authenticate the call and authorize access.

There are three primary participants in the OAuth process flow:

- **Client** – Wants access to a resource protected by a Resource Server and interacts with an Authorization Server to obtain access tokens
- **Resource Server (RS)** – Hosts and protects resources and makes them available to properly authenticated and authorized clients
- **Authorization Server (AS)** – Issues access tokens and refresh tokens to clients on behalf of Resource Servers

Tokens

- **Access Token** – Allows clients to authenticate to a resource server and claim authorizations for accessing particular resources. Access tokens have specific authorization scope and duration.
- **Refresh Token** – Allows clients to obtain a fresh access token without re-obtaining authorization from the resource owner. It is a long-lived token that a client can trade in to an authorization server to obtain a new access token (with the same attached authorizations as the existing access token).

The OAuth AS in PingFederate supports a wide variety of different interaction models appropriate for different types of clients such as a server, a desktop application, or an application on a phone or a tablet.



Note: The PingFederate OAuth AS implementation is based on the Internet Engineering Task Force (IETF) [OAuth 2.0 Authorization Framework](http://tools.ietf.org/html/rfc6749) (<http://tools.ietf.org/html/rfc6749>).

The following section describes the [Web Redirect Flow](#), which is the primary scenario for OAuth transactions.

Web Redirect Flow

In this scenario, a user attempts to access a protected resource through a third-party Web server client. The client sends an authorization request to the resource server and receives a code back via an HTTP redirect. The client trades the code for an access token, and then uses the token in a API call to obtain data.

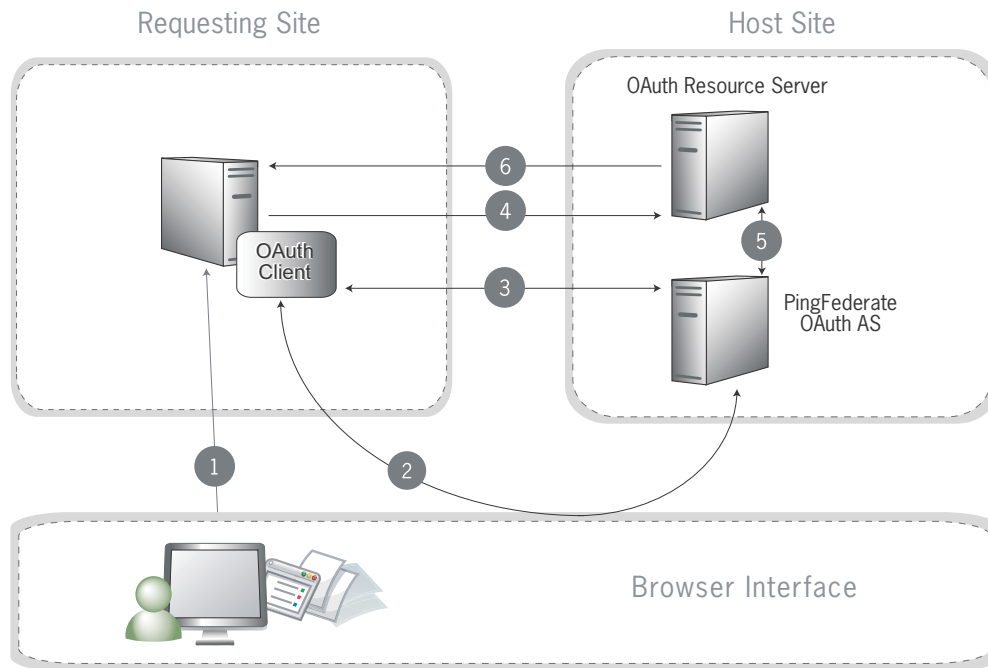


Figure 23: Web Redirect

Processing Steps

1. User navigates to an OAuth client Web site (the requesting site) and requests access to protected resources from another Web site.
2. The browser is redirected to the PingFederate OAuth AS with a request for authorization.

If the user is not already logged on, the OAuth AS challenges the user to authenticate. The OAuth AS authenticates the user and provides a consent page for the user to authorize the sharing of information. Once the user

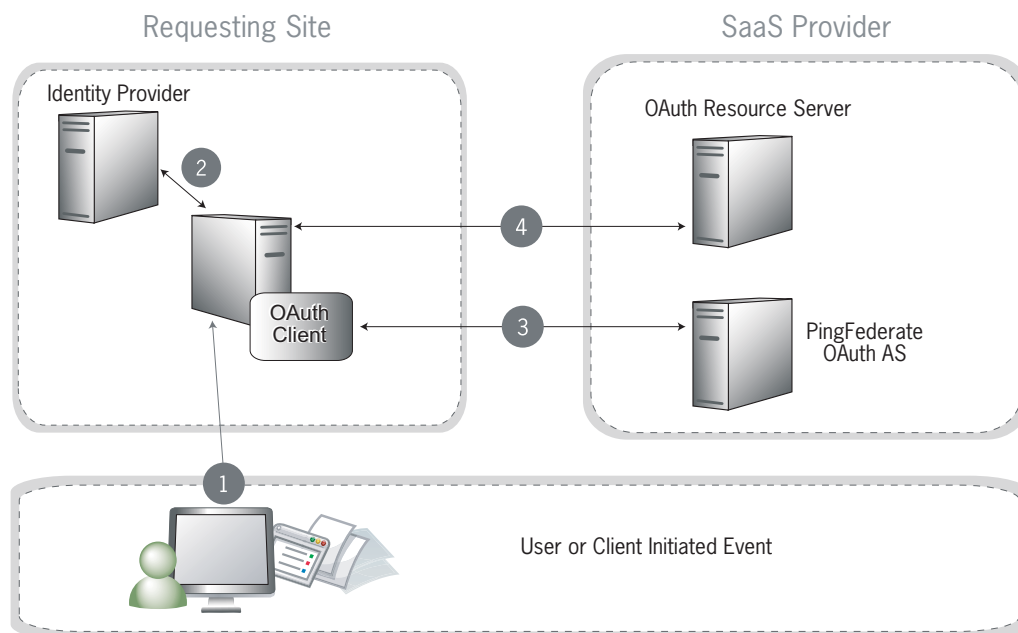
authorizes, the OAuth AS redirects the browser to the requesting site with an authorization code.

If the user does not authenticate, an error is returned rather than the authorization code.

3. The requesting site makes an HTTPS request to the OAuth AS to exchange the authorization code for an access token. OAuth AS validates the grant and user data associated with the code and then returns an access token.
4. The requesting site uses the access token in an API call to request user data.
5. The Resource Server asks PingFederate for verification that the token is valid and has not expired. PingFederate returns data about the user, the granted scope, and the client ID.
6. Once verified, the Resource Server returns the requested data to the requesting site.
7. Not shown. The requesting site displays data from the API call to the user.

SAML 2.0 Profile for OAuth 2.0 Authorization Grants

In this scenario, a client obtains a SAML 2.0 bearer assertion and makes an HTTP request to the PingFederate OAuth AS to exchange the SAML assertion for an access token. The AS validates the assertion and returns an access token. The client uses the token in an API call to the Resource Server to obtain data.



Processing Steps

1. Some user-initiated or client-initiated event (for example, a mobile application or a scheduled task) requests access to Software as a Service (SaaS) protected resources from an OAuth client application.

2. The client application obtains a SAML 2.0 bearer assertion from a local Identity Provider (IdP) for example, PingFederate.
3. The client makes an HTTP request to the PingFederate OAuth AS to exchange the SAML assertion for an access token. The AS validates the assertion and returns the access token. For more information on how the AS performs the validation, see the [specification](http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer) (<http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer>).
4. The client application adds the access token to its API call to the Resource Server. The Resource Server returns the requested data to the client.

OpenID Connect Support

As an extension of OAuth, PingFederate also supports Basic and Implicit Profiles defined for OpenID Connect, an emerging standard. For more information, see the [OpenID Connect Web site](http://openid.net/connect) (openid.net/connect).

System for Cross-domain Identity Management (SCIM)

PingFederate supports the SCIM 1.1 protocol for outbound as well as inbound provisioning. At an IdP (outbound) site, you can automatically provision and maintain user accounts at service-provider sites that have implemented SCIM. When PingFederate is configured as an SP (inbound), you can provision and manage user accounts and groups, for your own organization automatically using the standard SCIM protocol.

For more information regarding outbound provisioning for IdPs and inbound provisioning for SPs, see “[User Provisioning](#)” in the “Key Concepts” chapter of the PingFederate *Administrator’s Manual*. For detailed information about SCIM, see the Web site www.simplecloud.info.

Transport and Message Security

The standards generally define two main ways of securing interactions: Secure Sockets Layer with Transport Level Security (SSL/TLS) and digital signatures. SSL/TLS is used in environments where both message confidentiality and integrity are required. For SAML messaging, digital signatures are used to ensure the identity of both parties involved in the transaction and to validate that a message was received from a particular partner.

With PingFederate you can also choose to encrypt SAML 2.0 messages, including SAML metadata files, as well as WS-Trust STS assertions to achieve increased privacy.

For more information, refer to [Security and Privacy Considerations for the OASIS Security Assertion Markup Language \(SAML\) V2.0](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security#samlv20) available on the [SSTC Web site](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security#samlv20) (www.oasis-open.org/committees/tc_home.php?wg_abbrev=security#samlv20).

Using the Thales nShield Connect HSM

For optimal security, PingFederate can be configured to use a hardware security module (HSM) for cryptographic material storage and operations. Standards such as the Federal Information Processing Standard (FIPS) 140-2 require the storage and processing of all keys and certificates on a certified cryptographic module.

PingFederate is engineered and tested with the standard-compliant Thales nShield Connect (formerly nCipher Connect) HSM. The first step is to install and configure the Thales nShield Connect HSM according to the manufacturer's documentation. Once installed, use the information in the following sections to configure PingFederate to interact with the HSM for key generation, storage, and operation.

HSM Operational Notes

Some restrictions apply to the operation of PingFederate when using an HSM:

- Thales nShield Connect does not support Java 8. Install JDK 7 on the PingFederate server (see [“Hardware Security Module \(Optional\)”](#) on page 16 and [“Java Environment”](#) on page 14 for more information).
- As an OpenID Connect Provider, PingFederate generates and rotates temporary asymmetric key pairs to sign ID Tokens for Relying Parties. These in-memory short-term keys are not stored on the HSM. (For more information about OpenID Connect, see [“OAuth 2.0”](#) on page 57 and [“OpenID Connect Support”](#) on page 60.)
- Private keys are not exportable. When configured for use with the HSM, administrative-console options for this feature are disabled. Only the public portion of generated keys is exportable.
- When running in FIPS 140-2 level 3 compliance (also known as strict FIPS mode) private keys can not be imported. In this case administrative-console options for this feature are disabled.

- Not all cipher suites in a standard Java configuration are available. They are limited to those listed in the file named `com.pingidentity.crypto.NcipherJCEManager.xml` located in the `<pf_install>/pingfederate/server/default/data/config-store` directory.
- When using the Configuration Archive feature, any keys, certificates, or objects generated and stored on the HSM prior to saving a configuration archive must continue to exist unaltered when the archive is restored (see “Using the Configuration Archive Utility” in the “System Administration” chapter of the *PingFederate Administrator’s Manual*). In other words, any deletion or creation of objects on the HSM not executed via the PingFederate user interface will not be recognized or operational.

For example, during the course of normal PingFederate operation you create and save objects A, B and C to the HSM and create a data archive that contains references to those objects. If you then delete object C and attempt to recover it via the data archive, PingFederate will fail, producing various exceptions. Because the data archive contains a reference to the object and the object has been deleted from the HSM, it is not possible to use that data archive again.

nShield Installation and Configuration

To use PingFederate with the nShield Connect HSM:

1. Install and configure the nShield Connect HSM client software.
As part of the installation, install the optional Java Support (including KeySafe) and nCipherKM JCA/JCE provider classes components).
During installation, disregard any message about noncompliant Java versions. JDK 7 is required. Accept the remaining defaults when prompted by the installer.
2. After your installation, refer to the Thales nShield documentation to see how to make your PingFederate server a client of an HSM server.



Note: PingFederate currently supports only Operator Card Set (OCS) protected keys. Note the password used for the OCS; you will need the password for your installation of PingFederate.

3. If you have not already done so, download and install the (JCE) Unlimited Strength Jurisdiction Policy Files 7 (`UnlimitedJCEPolicyJDK7.zip`).
Follow instructions in the readme to install the Policy Files.
4. To enable the Java interface, copy the `nCipherKM.jar` file from the `NFAST_HOME\java\classes` folder into your `JAVA_HOME\jre\lib\ext` folder.

Thales provides some sample Java applications that may be run to ensure that the Java/HSM interface is working properly prior to installing PingFederate. Please refer to Thales documentation for more information.

5. In your Java SDK directory, open the file `java.security` in the `jre/lib/security` directory and add the **boldface** line below to the list of security providers, *after* all Sun providers:


```
# List of providers and their preference orders (see above):
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=com.sun.net.ssl.internal.ssl.Provider
security.provider.4=com.sun.crypto.provider.SunJCE
security.provider.5=sun.security.jgss.SunProvider
security.provider.6=com.sun.security.sasl.Provider
security.provider.7=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.8=sun.security.smartcardio.SunPCSC
security.provider.9=sun.security.mscapi.SunMSCAPI
security.provider.10=com.ncipher.provider.km.nCipherKM
```
6. Save and close the `java.security` file.
7. Install PingFederate on the network interconnected to the HSM (see [“Installation”](#) on page 13).
8. In the `<pf_install>/pingfederate/server/default/data` directory, delete files with the extension `.jks`, specifically:
 - `ping-dsig.jks`
 - `ping-ssl-server.jks`
 - `ping-ssl.jks`
 - `ping-trust.jks`
9. In the `hivemodule.xml` file in the `<pf_install>/pingfederate/server/default/conf/META-INF` directory, in the `Crypto` provider section, change the value of the `construct class`, as shown below in **bold**:


```
<construct
class="com.pingidentity.crypto.NcipherJCEManager" />
```
10. Below the code in the previous step, change the value of the `construct class` to indicate that nShield is used as the Certificate Service service point as shown below in **bold**:


```
<construct
class="com.pingidentity.crypto.NcipherCertificateServiceImpl" />
```
11. Save and close the `hivemodule.xml` file.
12. In the `run.properties` file found in the `<pf_install>/pingfederate/bin` directory, change the value of the `pf.hsm.mode` property near the end of this file from `OFF` to `NCIPHER`, as shown below:


```
pf.hsm.mode=NCIPHER
```
13. Save and close the `run.properties` file.

14. From the `<pf_install>/pingfederate/bin` directory, run the `hsmpass.bat` batch file for Windows or the `hsmpass.sh` script for UNIX/Linux.

Enter the Operator Card Set password when prompted (see [Step 2](#)).

This procedure sets and securely stores the password for communication to the HSM from PingFederate.

15. For clustered-server installations, see the next section.

This completes the steps required to configure PingFederate for use with nShield Connect. You may start the PingFederate server in the normal way and proceed as you would for any other installation (see [“Running PingFederate for the First Time”](#) on page 18).

Additional Steps for Server Clusters

If your PingFederate installation is configured in a clustered environment, use these steps to replicate nShield data to other connected nodes in the cluster.

1. In the administrative-console installation, locate the directory `<pf_install>/pingfederate/server/default/data` and create a directory named `ncipher-kmdata-local`.
2. Copy into `ncipher-kmdata-local` all files from the `NFAST_KMDATA\local` directory, where `NFAST_KMDATA` is an environment variable created during the nShield Connect installation.

For example, `NFAST_KMDATA` could be set to `C:\ProgramData\nCipher\Key Management Data`.
3. Create a new environment variable named `NFAST_KMLOCAL` and set it to `<pf_install>/pingfederate/server/default/data/ncipher-kmdata-local`.



Note: Perform this step on all servers within the cluster.

4. Restart the nShield Connect hardserver on all PingFederate servers in the cluster. (See the Thales documentation for instructions on restarting the hardserver.)
5. Use the administrative console to replicate the new configuration (see the PingFederate Server Clustering Guide).

Using the SafeNet Luna HSM

For optimal security, PingFederate can be configured to use a hardware security module (HSM) for cryptographic material storage and operations. Standards such as the Federal Information Processing Standard (FIPS) 140-2 require the storage and processing of all keys and certificates on a certified cryptographic module.

PingFederate is engineered and tested with the standard-compliant SafeNet Luna SA HSM. The first step is to install and configure the Luna SA HSM according to the manufacturer's documentation. Once installed, use the information in the following sections to configure PingFederate to interact with the HSM for key generation, storage, and operation.

Operational Notes

Some restrictions apply to PingFederate operations when using a Luna SA HSM:



Note: For Safenet Luna SA HSM versions 5.x, PingFederate does not store public certificates (for both signature and encryption) on the hardware module. In this case, certificates are stored in keystores located on the file system.

- Luna SA does not support Java 8. Install JDK 7 on the PingFederate server (see [“Hardware Security Module \(Optional\)”](#) on page 16 and [“Java Environment”](#) on page 14 for more information).
- As an OpenID Connect Provider, PingFederate generates and rotates temporary asymmetric key pairs to sign ID Tokens for Relying Parties. These in-memory short-term keys are not stored on the HSM. (For more information about OpenID Connect, see [“OAuth 2.0”](#) on page 57 and [“OpenID Connect Support”](#) on page 60.)
- Private keys are not exportable. When configured for use with the HSM, administrative-console options for this feature are disabled. Only the public portion of generated keys is exportable.

- Not all cipher suites in a standard Java configuration are available. They are limited to those listed in the file named `com.pingidentity.crypto.LunaJCEManager.xml` located in the `<pf_install>/pingfederate/server/default/data/config-store` directory.
- When using the Configuration Archive feature, any keys, certificates, or objects generated and stored on the HSM prior to saving a configuration archive must continue to exist unaltered when the archive is restored (see [“Using the Configuration Archive Utility”](#) in the “System Administration” chapter of the *PingFederate Administrator’s Manual*).

For example, during the course of normal PingFederate operation you create and save objects A, B, and C to the HSM and create a data archive that contains references to those objects. If you then delete object C and attempt to recover it via the data archive, PingFederate fails, producing various exceptions. Because the data archive contains a reference to the object and the object has been deleted from the HSM, it is not possible to use that data archive again.

Installation and Configuration

To use PingFederate with the Luna SA HSM:

1. Install and configure your SafeNet Luna SA HSM, including the optional package for Java (referred to as the JSP), according to SafeNet’s instructions.

This includes the creation of a partition, creation of a Network Trust Link (NTL), and assignment of a client to a partition. Ensure that you can perform the `vtl verify` command indicating that you are communicating securely and properly to the HSM.

Delete any unnecessary keys or objects that may have been created while testing communication to the HSM from the host that runs PingFederate.

Note the password used to open communication to the HSM via the NTL. You need this for your PingFederate installation.

2. To enable the Java interface, copy the following files to your Java installation:

For Windows:

- Copy the `LunaAPI.dll` file either to an arbitrary folder and add the folder’s path as a system variable or to the Windows system folder in the `C:\Windows\System32` directory.
- Copy these files from `LUNA_HOME\jsp\lib` into your `JAVA_HOME\jre\lib\ext` folder:

Luna 4.x: `LunaJCASP.jar` and `LunaJCESP.jar`

Luna 5.x: `LunaProvider.jar`

For UNIX/Linux:

- Copy these files from `LUNA_HOME/jsp/lib` into your `JAVA_HOME/jre/lib/ext` folder:

`libLunaAPI.so`

Luna 4.x: `LunaJCASP.jar` and `LunaJCESP.jar`

Luna 5.x: `LunaProvider.jar`

SafeNet provides some sample Java applications that may be run to ensure that the Java/HSM interface is working properly prior to installing PingFederate. Please contact SafeNet support for more information.

3. In your Java SDK directory, open the file `java.security` in the `jre/lib/security` directory and add the line in **boldface** below to the list of security providers, *immediately before* the `sun.security.ec.SunEC` providers:

Luna 4.x:

```
# List of providers and their preference orders (see above):
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=com.chrysalisits.crypto.LunaJCAProvider
security.provider.4=com.chrysalisits.cryptox.LunaJCEProvider
security.provider.5=sun.security.ec.SunEC
security.provider.6=com.sun.net.ssl.internal.ssl.Provider
security.provider.7=com.sun.crypto.provider.SunJCE
security.provider.8=sun.security.jgss.SunProvider
security.provider.9=com.sun.security.sasl.Provider
security.provider.10=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.11=sun.security.smartcardio.SunPCSC
```

Luna 5.x:

```
# List of providers and their preference orders (see above):
security.provider.1=sun.security.provider.Sun
security.provider.2=sun.security.rsa.SunRsaSign
security.provider.3=com.safenetinc.luna.provider.LunaProvider
security.provider.4=sun.security.ec.SunEC
security.provider.5=com.sun.net.ssl.internal.ssl.Provider
security.provider.6=com.sun.crypto.provider.SunJCE
security.provider.7=sun.security.jgss.SunProvider
security.provider.8=com.sun.security.sasl.Provider
security.provider.9=org.jcp.xml.dsig.internal.dom.XMLDSigRI
security.provider.10=sun.security.smartcardio.SunPCSC
```

4. Save and close the `java.security` file.
5. Install PingFederate on the network interconnected to the HSM (see [“Installation”](#) on page 13).
6. In the `<pf_install>/pingfederate/server/default/data` directory, delete files with the extension `.jks`, specifically:
 - `ping-dsig.jks`
 - `ping-dsig-cert.jks`
 - `ping-ssl-server.jks`
 - `ping-ssl-server-cert.jks`
 - `ping-ssl.jks`
 - `ping-trust.jks`
 - `ping-ssl-client-trust-cas.jks`
7. In the `hivemodule.xml` file in the `<pf_install>/pingfederate/server/default/conf/META-INF` directory, in the `Crypto`

provider section, change the value of the construct class, as shown below in **bold**:

Luna 4.x:

```
<construct class="com.pingidentity.crypto.LunaJCEManager" />
```

Luna 5.x:

```
<construct class="com.pingidentity.crypto.LunaJCEManager5" />
```

8. Below the code in the previous step, change the value of the construct class to indicate that Luna is used as the Certificate Service service point, as shown below in **bold**:

Luna 4.x:

```
<construct
class="com.pingidentity.crypto.LunaCertificateServiceImpl" />
```

Luna 5.x:

```
<construct
class="com.pingidentity.crypto.LunaCertificateServiceImpl5" /
>
```

9. Save and close the `hivemodule.xml` file.
10. In the `run.properties` file found in the `<pf_install>/pingfederate/bin` directory, change the value of the `pf.hsm.mode` property near the end of this file from OFF to LUNA, as shown below in **bold**:
`pf.hsm.mode=LUNA`
11. Save and close the `run.properties` file.
12. From the `<pf_install>/pingfederate/bin` directory, run the `hsmypass.bat` batch file for Windows or the `hsmypass.sh` script for UNIX/Linux.

Enter the NTL password when prompted (see [Step 1](#)).

This procedure sets and securely stores the password for NTL communication to the HSM from PingFederate.



Note: The Luna SA HSM may be configured in a high-availability group—to do so, please refer to the SafeNet distributed-installation instructions. To properly synchronize data, ensure that the `HAOnly` property is enabled using this command:

```
vtl haAdmin -HAOnly -enable
```

This completes the steps required to configure PingFederate for use with the Luna SA. You may start the PingFederate server in the normal way and proceed as you would for any other installation (see [“Running PingFederate for the First Time”](#) on page 18).

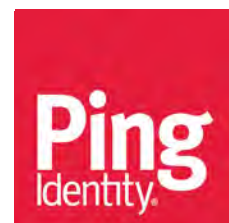


Important: To ensure expected behavior, SafeNet recommends restarting dependent processes such as PingFederate (including all server nodes in a cluster) whenever the Luna HSM is restarted.

PingFederate®

Version 7.3

Administrator's Manual



© 2005-2015 Ping Identity® Corporation. All rights reserved.

PingFederate *Administrator's Manual*
Version 7.3
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909
Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to the online documentation at documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change on this date: **January 19, 2017**.

Contents

	Preface	1
	About This Manual	1
	Overview	1
	Intended Audience	2
	Text Conventions	2
	Other Documentation	3
Chapter 1	Key Concepts	5
	Connection Types	5
	About WS-Trust STS	6
	Connection-Based Policy	6
	IdP Configuration	6
	SP Configuration	7
	Token Processors and Generators	7
	Bundled Token Plug-ins	7
	Commercial Token Plug-ins	8
	WSC and WSP Support	8
	Client SDK	8
	Windows Identity Foundation Clients	8
	STS OAuth Integration	9
	About OAuth	10
	Delegated Access Types	10
	Token Models and Management	10
	Token Security Model	10
	Token Data Model	11
	Token Management	11
	Grant Types	11
	Primary Grant Types	11
	Extension Grant Types	12

Persistent vs. Transient Grants	12
Mapping OAuth Attributes	13
The First Stage	14
The Second Stage	14
Runtime Processing	14
Scopes	14
OAuth User-Facing Screens	14
OpenID Connect	15
SSO Integration Kits and Adapters	15
Bundled Adapters	16
Bundled Authentication Selectors	16
Commercial Adapters and Selectors	17
Software Development Kit	18
Hierarchical Plug-in Configurations	18
Identity Mapping	18
Account Linking	19
Linking Permission and “Defederation”	19
SP Affiliations	19
Account Mapping	20
About Attributes	20
Attribute Contracts	21
Name Formats	22
STS Namespaces	22
Adapter Contracts	22
Extended Adapter Contract	23
STS Token Contracts	23
Extended Token Generator Contract	23
Data Stores	23
Multiple Data Source Attribute Mapping	24
Attribute Masking	25
About Token Authorization	25
Issuance Criteria	26
Single and Multi-Value Conditions	26
Security Infrastructure	27
Digital Signatures	27
Message Signing	27
Certificate Validation	28
Digital Signing Policy Coordination	29
Secure Sockets Layer	30
SAML SSL and TLS Scenarios	30
Authentication	30
Verifying Trusted Certificates	31
XML Encryption	31
Using Auto-Connect	32
Providing Metadata	32

	Runtime Processing	32
	Auto-Connect Security Model	34
	User Provisioning	34
	Outbound Provisioning for IdPs	35
	Provisioning for SPs	36
	Inbound Provisioning	36
	Just-in-Time Provisioning	36
	Federation Planning Checklist	37
	Multiple Virtual Server IDs	39
	Connecting to a Partner in One Connection	39
	Connecting to a Partner in Multiple Connections	39
	Working with Multiple Virtual Server IDs	40
	Configuration Data Exchange	40
	IdP to SP	41
	SP to IdP	41
	Mutual Settings Between Parties	41
	Federation Hub	42
	Bridging an IdP to an SP	42
	Bridging an IdP to multiple SPs	43
	Bridging multiple IdPs to an SP	44
	Bridging multiple IdPs to multiple SPs	45
	Federation Hub and Virtual Server IDs	47
Chapter 2	System Administration	49
	Starting and Stopping PingFederate	50
	Managing Log Files	50
	Administrator Audit Logging	52
	Administrator API Audit Logging	53
	Runtime Transaction Logging	53
	Transaction Logging Modes	54
	Security Audit Logging	54
	Outbound Provisioning Audit Logging	56
	Server Logging	57
	Using the Server Log Filter	57
	Writing Logs to Other Formats	59
	Writing Logs to Databases	59
	Writing Audit or Provisioner Audit Logs to CEF	61
	Writing Audit Logs for Splunk	62
	Exporting Metadata	63
	Choosing the Metadata Export Mode	65
	Defining a Metadata Attribute Contract	65
	Including a Signing Key	66
	Signing Metadata	66
	XML Encryption Certificates	67
	Completing the Export	67
	Signing XML Files	67

Using the Configuration Archive Utility	68
Using the Archive Import Screen	70
Account Management	71
Setting or Resetting Passwords	73
Changing Passwords	74
Alternative Console Authentication	74
Using LDAP Authentication	75
Using RADIUS Authentication	75
Using Certificate-Based Authentication	76
Managing Email Configuration	77
Using Virtual Host Names	79
Changing Configuration Parameters	80
Installing a New License Key	83
Automating Configuration Migration	84
Copying the Key from the Source to the Target Server	85
Administrative Console Migration	85
Using the Migration Tool	86
Outbound Provisioning CLI	89
Customizing User-Facing Screens	93
IdP User-Facing Pages	94
SP User-Facing Pages	96
Either IdP or SP User-Facing Pages	97
OAuth User-Facing Pages	98
Localization	98
Overriding Locales with Cookies	99
Retrieving Localized Messages	99
Configuring Password Policy	100
Extending the Lifetime of the PF Cookie	100
Adding Custom HTTP Response Headers	101
Customizing the Favicon for Application and Protocol Endpoints	101
Chapter 3 System Settings	103
Managing Server Settings	103
Setting Administration Options	104
Entering System Information	105
Configuring Runtime Notifications	105
Configuring Runtime Reporting	106
Using SNMP Monitoring	106
Runtime Monitoring Using JMX	107
Managing Accounts	111
Choosing Roles and Protocols	112
Specifying Federation Information	114
Setting System Options	116
Disabling Automatic Connection Validation	116
Defining Data-Store Validation Intervals	117
Defining proxy options	117
Managing System Updates	118

	Configuring Outbound Provisioning Settings	119
	Configuring Auto-Connect Metadata Signing	120
	Configuring Auto-Connect Metadata Lifetime	121
	Saving and Editing Server Settings	122
	Managing Data Stores	122
	Configuring a JDBC Database Connection	123
	Setting Advanced Options	126
	Configuring an LDAP Connection	127
	Defining an LDAP Type	129
	Setting Advanced LDAP Options	130
	Specifying LDAP Binary Attributes	132
	Configuring a Custom Data Store	133
	Configuring a Custom Data Store Instance	133
	Adapter Actions	134
	Editing and Saving a Data Store	134
	Defining an Account-Linking Data Store	134
	Defining an OAuth Grant Data Store	136
	Defining an OAuth Client Data Store	137
	Configuring IdP Discovery	139
	Standard IdP Discovery	139
	Choosing Domain Cookie Settings	139
	Configuring a Common Domain Service	140
	Configuring a Local Common Domain Server	141
	Editing and Saving the Configuration	142
	IdP Discovery Using a Persistent Cookie	142
	Configuration	142
	Configuring Redirect Validation	143
	Managing Partner Redirect Validation	145
Chapter 4	IdP-to-SP Bridging	147
	Adapter-to-Adapter Mapping	147
	Managing Mappings	148
	Assigning a License Group	148
	Configuring Attribute Lookup for Adapter-to-Adapter Mapping	149
	Choosing a Data Store (Optional)	149
	Identifying Target Application (Optional)	151
	Adapter Contract Fulfillment	151
	Configuring a Default Target URL (Optional)	152
	Configuring Issuance Criteria (Optional)	153
	Using the Summary Screen	155
	Token Exchange Mapping	155
	Managing Token Mappings	155
	Configuring Attribute Lookup for Token Exchange Mapping	156
	Choosing a Data Store for Token Mapping	157
	Token Contract Fulfillment	158
	Selecting Token Issuance Criteria	159
	Using the Token Mapping Summary Screen	162

	Connection Mapping Contracts	162
	Managing Contracts	162
	Editing Contract Info	163
	Defining Contract Attributes	164
	Using the Connection Mapping Contracts Summary Screen	164
Chapter 5	OAuth Configuration	165
	Enabling the OAuth AS	165
	Using OAuth Menu Selections	166
	Centralized Session Management	167
	Asynchronous Front-Channel Logout	167
	Back-Channel Session Revocation	168
	Authorization Server Settings	169
	Access Token Management	175
	Managing Access Token Management Instances	176
	Defining an Access Token Management Instance	176
	Configuring a Token-Management Instance	177
	Configuring Reference-Token Management	177
	Configuring JSON-Token Management	179
	Defining the Access Token Attribute Contract	182
	Configuring Resource URIs	182
	Defining Access Control	183
	Using the Access Token Management Summary Screen	183
	Configuring OpenID Connect Policies	184
	Configuring the Policy Attribute Contract	185
	Policy Attribute Sources and User Lookup	185
	Policy Contract Fulfillment	186
	Identifying Issuance Criteria for Policy Mapping	188
	Using the Policy Summary Screen	190
	Client Management	190
	Configuring a Client	191
	Resource-Owner Credentials Mapping	195
	Resource-Owner Attribute Sources and User Lookup	196
	Resource-Owner Contract Fulfillment	197
	Identifying Issuance Criteria for Credentials Mapping	198
	Using the Credentials-Mapping Summary Screen	200
	IdP Adapter Mapping for OAuth	200
	IdP Adapter Attribute Sources and User Lookup	200
	Grant Contract Fulfillment	201
	Defining Issuance Criteria for OAuth Adapter Mapping	202
	Using the Adapter-Mapping Summary Screen	204
	Access Token Mapping	204
	Access Token Attribute Sources and User Lookup	205
	Token Attribute Contract Fulfillment	206
	Configuring Issuance Criteria for Access Token Mapping	207
	Using the Token-Mapping Summary Screen	209

	Configuring an OAuth SAML Grant IdP Connection	209
	Specifying an Attribute Contract for the OAuth SAML Grant	210
	Configuring Access Token Manager Mappings	211
	Selecting an Access Token Manager Instance	211
	Choosing a Data Store for OAuth SAML Grant Attribute Mapping	211
	OAuth SAML Grant Contract Fulfillment	212
	Selecting Issuance Criteria for OAuth SAML Grant	214
	OAuth SAML Grant Attribute Mapping Configuration Summary	216
	OAuth SAML Grant Configuration Summary	216
	OAuth Attribute Mapping Using a Data Store	216
	Defining a JDBC Location for OAuth	217
	Configuring an OAuth Database Filter (WHERE Clause)	218
	Searching LDAP for OAuth Mapping	218
	Configuring an LDAP Filter for OAuth Mapping	219
	Configuring OAuth Custom Source Filters	220
	Selecting OAuth Custom Source Fields	220
Chapter 6	Security Management	221
	Certificate Management	221
	Trusted Certificate Authorities	222
	SSL Server Certificates	222
	SSL Client Keys and Certificates	225
	Digital Signing and Decryption Keys and Certificates	227
	Certificate Revocation Checking	230
	Authentication	233
	Application Authentication	234
	Validating Password Credentials	235
	Choosing a Type	236
	Configuring the LDAP Credential Validator	237
	Configuring the Simple Credential Validator	239
	Configuring the RADIUS Credential Validator	240
	Configuring the PingOne Directory Credential Validator	242
	Extending Contracts for the Credential Validator	243
	Finishing the Validator Configuration	244
	Using AD Domains and Kerberos Realms	244
	Configuring a Domain or Realm	245
	Managing Domain or Realm Settings	246
Chapter 7	Identity Provider SSO Configuration	249
	Application Integration Settings	250
	Configuring IdP Adapters	250
	Selecting an IdP Adapter Type	251
	Configuring an IdP Adapter	252
	Invoking Adapter Actions	253
	Extending an Adapter Contract	253
	Setting Pseudonym Values and Masking	254
	Selecting an Authentication Context	255

Editing and Saving Adapter Instances	255
Configuring Authentication Selectors	255
Choosing a Selector Type	256
Configuring the CIDR Authentication Selector	257
Configuring the Cluster Node Authentication Selector	257
Defining Cluster Node Results	258
Configuring the Connection Set Authentication Selector	258
Configuring the HTTP Header Authentication Selector	259
Configuring the OAuth Scope Authentication Selector	260
Configuring the Requested AuthN Context Authentication Selector	261
Defining AuthN Context Results	261
Finishing the Selector-Instance Configuration	262
Mapping Selector Results to Authentication Sources	262
Configuring a Default URL and Error Message	264
Viewing Application Endpoints	265
Viewing Protocol Endpoints	265
Managing SP Connections	267
Accessing Connections	268
Via the Main Menu	268
Via the Manage Connections Screen	269
Choosing a Connection Template	273
Choosing a Connection Type	273
Choosing Connection Options	275
Importing Metadata	275
Importing a Verification Certificate	276
Viewing the Metadata Summary	276
General Information	276
Configuring Browser-Based SSO	278
Configuration Steps	278
Choosing Profiles (SAML 2.0)	280
Setting an Assertion Lifetime	282
Assertion Creation	283
Configuring Protocol Settings	313
Editing and Saving Browser SSO Settings	322
Configuring the Attribute Query Profile	322
Defining Retrievable Attributes	323
Configuring Attribute Lookup	323
Choosing a User-Data Store	324
Configuring Data Store Lookup	325
Attribute Mapping Fulfillment	326
Defining Issuance Criteria (Optional)	327
Specifying Security Policy	329
Editing and Saving Attribute Query Configurations	330
Configuring Credentials	330
Configuring Back-Channel Authentication	330

	Configuring Digital Signature Settings	333
	Configuring Signature Verification Settings	334
	Selecting an Encryption Certificate (SAML)	337
	Selecting a Decryption Key (SAML)	338
	Editing and Saving Credential Configurations	339
	Configuring Outbound Provisioning	339
	Defining a Provisioning Target	340
	Managing Channels	342
	Specifying Channel Information	343
	Identifying the Source Data Store	343
	Modifying Source Settings	344
	Specifying a Source Location	347
	Mapping Attributes	348
	Channel Activation and Summary	352
	Editing and Activating a Connection	353
	Defining SP Affiliations	354
	Using the Manage Affiliations Screen	354
	Importing Affiliation Metadata	355
	Entering Affiliation Information	355
	Managing Affiliation Membership	355
	Activating and Editing the Affiliation	356
	Configuring SP Auto-Connect	356
	Initial Setup	357
	Specifying an Assertion Lifetime	357
	Choosing a Signing Certificate	357
	Configuring Assertion Creation	358
	Auto-Connect Activation and Summary	358
	Specifying Allowed SP Domains	359
Chapter 8	Service Provider SSO Configuration	361
	SP Application Integration Settings	362
	Configuring SP Adapters	362
	Creating an Adapter Instance	363
	Configuring an Adapter Instance	364
	Choosing Adapter Actions	365
	Extending Adapter Contracts	365
	Editing and Saving SP Adapter Instances	366
	Configuring Target URL Mapping	366
	Configuring Identity Store Provisioners	368
	Creating an Identity Store Provisioner Instance	368
	Defining Identity Store Provisioner Behavior	370
	Extending Identity Store Provisioner Contracts	370
	Extending Identity Store Provisioner Contracts for Groups	370
	Editing and Saving Identity Store Provisioner Instances	371
	Configuring Default URLs	372
	Viewing SP Application Endpoints	372

Federation Settings	372
Attribute Requester Mapping	373
Viewing SP Protocol Endpoints	374
Managing IdP Connections	376
Accessing IdP Connections	377
From the Main Menu	377
From the Manage Connections Screen	378
Choosing an IdP Connection Type	381
Choosing IdP Connection Options	382
Importing IdP Metadata	383
Importing a Certificate	383
Viewing the Summary	383
General Connection Information	383
Configuring Browser SSO	385
Connection Configuration Steps	386
Choosing SAML Profiles	387
User-Session Creation	390
Configuring OAuth Attribute Mapping	411
Configuring SAML Protocol Settings	415
Editing and Saving SSO Settings	425
Configuring the Attribute Query Option	425
Setting the Attribute Authority Service URL	425
Mapping Attribute Names	426
Defining Security Policy	427
Saving the Attribute Query Configuration	427
Using Just-in-Time Provisioning	427
Selecting Attribute Sources (SAML 2.0)	428
Identifying the User Repository	429
Specifying an LDAP User-Record Location	429
Defining an LDAP Filter	430
Identifying Provisioning Attributes for LDAP	431
Selecting a SQL Method	431
Specifying a Database User-Record Location	432
Specifying a Unique-ID Database Column	433
Specifying a Stored-Procedure Location	433
Mapping Attributes to a User Account	434
Choosing an Event Trigger	437
Error Handling	437
Using the Provisioning Summary Screen	438
Configuring Inbound Provisioning	438
Specifying the User Repository	439
Identifying an LDAP User-Record Location	439
Defining a Unique ID	439
Defining a Unique Group ID	440
Writing User Information to the Data Store	441

	Configuring a SCIM Response	444
	Handling SCIM Delete Requests	448
	Writing Group Information to the Data Store	448
	Configuring a SCIM Response for Groups	451
	Saving the Inbound Provisioning Configuration	454
	Configuring Security Credentials	454
	Back-Channel Authentication	454
	Digital Signature Settings	457
	Signature Verification Settings	458
	Choosing an Encryption Certificate	461
	Choosing a Decryption Key	462
	Saving Credential Configurations	463
	IdP Connection Activation and Summary	463
	Configuring IdP Auto-Connect	464
	Configuring the Initial Setup	464
	Choosing a Certificate	465
	Configuring User-Session Creation	465
	Connection Activation and Summary	466
	Specifying Allowed IdP Domains	467
Chapter 9	WS-Trust STS Configuration	469
	Server Settings	469
	Enabling the WS-Trust STS	469
	Configuring STS Authentication	470
	Selecting Authentication Methods	471
	Configuring Basic Authentication	471
	Configuring Mutual SSL Authentication	473
	Using the STS Summary Screen	475
	IdP Configuration for STS	475
	Configuring Token Processors	475
	Selecting a Token Processor Type	477
	Configuring a SAML Token Processor Instance	477
	Configuring an OAuth Token Processor Instance	479
	Configuring a JSON Web Token (JWT) Processor Instance	479
	Configuring a Username Token Processor Instance	480
	Extending a Processor Contract	481
	Setting Attribute Masking	482
	Editing and Saving Processor Instances	483
	Managing STS Request Parameters	483
	Creating a Request Contract	483
	Configuring SP Connections for STS	484
	Configuring IdP Protocol Settings	485
	Setting a Token Lifetime	486
	Configuring Token Creation	487
	SP Configuration for STS	509
	Configuring Token Generators	509

	Selecting a Token Generator Type	510
	Configuring a Token Generator Instance	510
	Extending a Generator Contract	512
	Editing and Saving Generator Instances	512
	Configuring IdP Connections for STS	513
	Configuring STS Protocol Settings	513
	Configuring Token Generation	514
Appendix A	OpenToken Adapter Configuration	533
	Configuring the IdP OpenToken Adapter	534
	Configuring the SP OpenToken Adapter	538
Appendix B	HTTP Basic Adapter Configuration	543
	Configuring the HTTP Basic IdP Adapter	543
Appendix C	HTML Form Adapter Configuration	549
	Configuring the HTML Form IdP Adapter	550
Appendix D	Composite Adapter Configuration	557
	Configuring the Composite Adapter	557
Appendix E	Application Endpoints	563
	IdP Endpoints	563
	SP Endpoints	567
	SP Services	567
	SCIM Inbound Provisioning Endpoints	572
	System-Services Endpoints	576
Appendix F	OAuth 2.0 Endpoints	579
	Token Endpoint	579
	Client Identification and Authentication	579
	Grant Type Parameters	580
	Authorization Code Grant Type	580
	Refresh Token Grant Type	581
	Resource Owner Credentials (Password) Grant Type	581
	Client Credentials Grant Type	582
	SAML 2.0 Bearer Assertion Grant Type	582
	Access Token Verification/Validation Grant Type	583
	Access Token Management Parameters	583
	Token Revocation Endpoint	585
	Client Identification and Authentication for Token Revocation	585
	Parameters	585
	Authorization Endpoint	586
	Grant-Management Endpoint	589
	OpenID Connect Metadata Endpoint	590
Appendix G	Web Service Interfaces	591
	Connection Management Service	592
	Exporting a Connection	592
	Exporting Manually	592
	Using the Connection Service	593
	Code Sample	593

	Importing Connections	593
	Sample Code	594
	Deleting Connections	594
	Code Example	594
	Cluster Configuration Replication	594
	Example Code	595
	Validation Disclaimer	595
	SSO Directory Service	595
	Coding Example	596
	SOAP Request and Response Example	597
	OAuth Client Management Service	598
	Endpoints	599
	/oauth/clients	599
	/oauth/clients/id	605
	OAuth Access Grant Management Service	606
	Session Revocation API	607
	PingFederate Administrative API	610
	Using the API Interactive Documentation	610
Appendix H	Using Attribute Mapping Expressions	613
	Enabling and Disabling Expressions	613
	Constructing Expressions	614
	Data Store Syntax	615
	Issuance Criteria Syntax	615
	Expression Examples	616
	Issuance Criteria and Multiple Virtual Server IDs	617
	Using the OGNL Edit Screen	618
Appendix I	Troubleshooting	621
	Data Store Issues	621
	Installation Issues	622
	Runtime Issues	622
	Server Startup	622
	Glossary	623
	List of Acronyms	629

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 5, “OAuth Configuration”](#) — How to configure PingFederate to act as an OAuth Authorization Server.
- [Chapter 6, “Security Management”](#) — Information about importing, exporting, and maintaining certificates and keys in PingFederate.
- [Chapter 7, “Identity Provider SSO Configuration”](#) — How to configure PingFederate to act as an Identity Provider (IdP) and establish connections to Service Providers.
- [Chapter 8, “Service Provider SSO Configuration”](#) — How to configure PingFederate to act as a Service Provider (SP) and establish connections to Identity Providers.
- [Chapter 9, “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.

- [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 F, “OAuth 2.0 Endpoints”](#) — Detailed information about using PingFederate connection endpoints for the OAuth Authorization Server.
- [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 71).

Text Conventions

This document uses the text conventions identified below.

Table 1: Text Conventions

Convention	Description
Fixed Width	Indicates text that must be typed exactly as shown in the instructions. Also used to represent program code, file names, and directory paths.
Blue text	Indicates hypertext links.
<i>Italic</i>	Used for emphasis and document titles.
▶ [text]	Used for procedures where only one step is required.

Table 1: Text Conventions (Continued)

Convention	Description
Sans serif	Identifies descriptive text on a user-interface screen. Example: “Print Document dialog”
Sans serif bold	Identifies menu items, navigational links, or buttons. For example: Click Save .

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.

Key Concepts

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 15
- [“Hierarchical Plug-in Configurations”](#) on page 18
- [“Identity Mapping”](#) on page 18
- [“About Attributes”](#) on page 20
- [“Security Infrastructure”](#) on page 27
- [“Using Auto-Connect”](#) on page 32
- [“User Provisioning”](#) on page 34
- [“Federation Planning Checklist”](#) on page 37
- [“Federation Hub”](#) on page 42



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 34).

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 15).

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 “[Token Exchange Mapping](#)” on page 155).

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 18).

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, JWT tokens, and Username tokens (see “[Token Models and Management](#)” 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.)

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](http://www.pingidentity.com/support-and-downloads) 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](http://www.pingidentity.com/support-and-downloads) (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](http://msdn.microsoft.com/en-us/library/ms730879.aspx) (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 577).
- Metadata Exchange, which contains connection details relating to the SP partner (see [“/pf/sts_mex.ping”](#) on page 577).

For more information about claim-based federated identity, see Microsoft's [A Guide to Claims-based Identity and Access Control](http://msdn.microsoft.com/en-us/library/ff423674.aspx) (msdn.microsoft.com/en-us/library/ff423674.aspx).

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 479 and [“Defining the Access Token Attribute Contract”](#) on page 182).

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](http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer) 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

```
(oauth-v2:access:token:response
|via|urn:ietf:params:oauth:grant-type:saml2-bearer)
```

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.)

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 411). 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 165 for more information.

For an STS IdP, an OAuth token processor is provided with the PingFederate installation (see “[IdP Configuration for STS](#)” on page 475).

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 and Management

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 security model and data model.

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.

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.



Note: PingFederate supports different token models through a plug-in architecture (see “[Access Token Management](#)” on page 175).

Token Management

PingFederate supports multiple access token management instances, providing flexibility for enterprises where deployments may require different token lifetimes, attribute contracts, and/or token validation rules for various clients.

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 579).

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.



Note: The SAML assertion used for this grant type generally cannot be a browser-based SSO assertion. To ensure its validity, the assertion must be associated with WS-Trust STS processing (see “[STS OAuth Integration](#)” on page 9).

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 revoked. Persistent grants can result from:

- The Refresh Token grant type when used in conjunction with either the Authorization Code or Resource Owner Password Credentials grants (see “[Configuring a Client](#)” on page 191).
- The Implicit grant type when the Reuse Existing Persistent Access Grants for Grant Types checkbox is selected (see “[Authorization Server Settings](#)” on page 169).

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

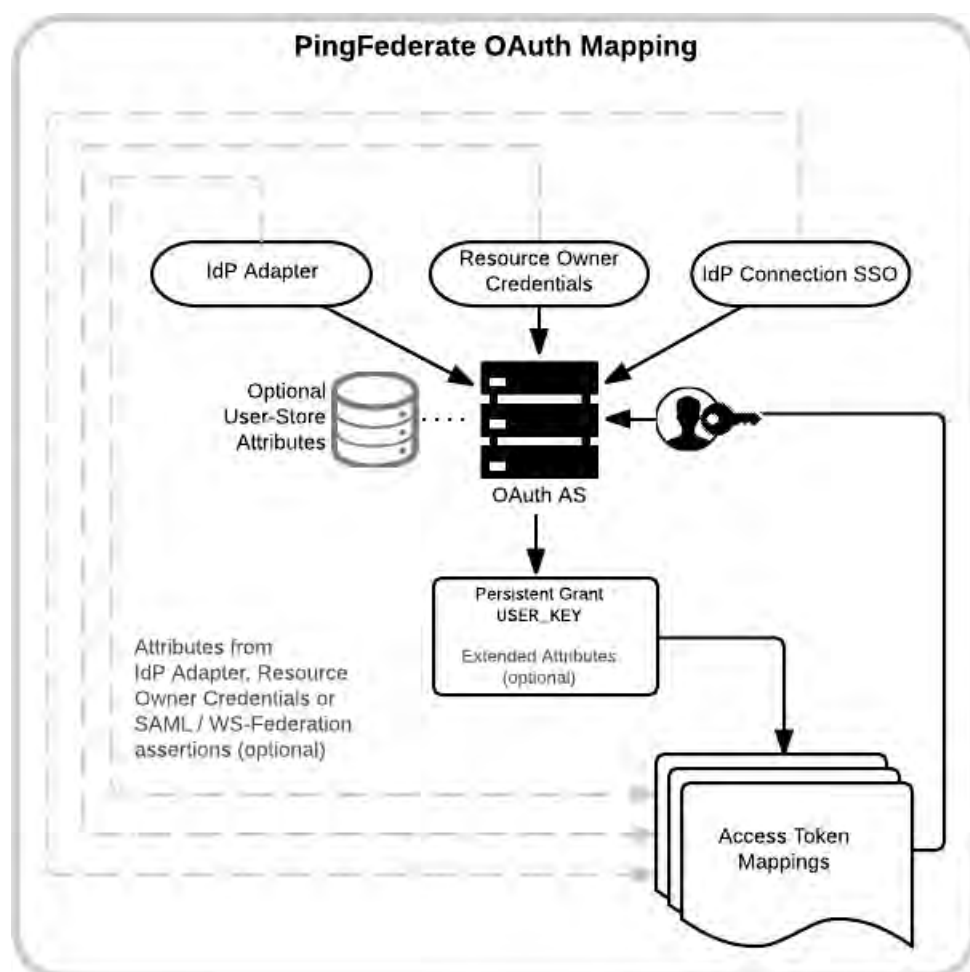
user's initial authentication verification within PingFederate (see [“Mapping OAuth Attributes”](#), next).

✓ 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 136 for more information.

Persistent grants also support Extended Attributes, where attributes obtained during a user's initial authentication can be used to map values into the access tokens (see [“Authorization Server Settings”](#) on page 169 and [“Mapping OAuth Attributes”](#), next).

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 two-stage mapping work flow is different from other mapping scenarios in PingFederate, which involve just a one-phase configuration—see [“About Attributes”](#) on page 20).



The First Stage

To accomplish the first stage (top of diagram in solid lines), mapping is required for setting up persistent grants, including a user key and all extended attributes (see the previous section, “[Persistent vs. Transient Grants](#)”).

The mappings may use attributes obtained during initial authentication events within PingFederate—via SAML or WS-Federation assertions (for Service Providers’ IdP connections for SSO), IdP adapters, and/or Resource Owner credentials (see “[Using OAuth Menu Selections](#)” on page 166). Data-store lookup is provided as needed (for example, to retrieve an LDAP ID for storage as the user key).



Note: The `USER_KEY` values must be unique across all end users because it is the identifier to store and to retrieve persistent grants. For example, if you have two Active Directory domains, the `sAMAccountName` value of an end user in one domain may collide with that of another end user in the other domain. In this scenario, you can map Subject DN to the `USER_KEY`.

The Second Stage

The second mapping configuration involves mapping from persistent grants (the user keys and/or any extended attributes derived from the first stage) into OAuth access tokens.

If the authentication context matches a specific mapping, then attributes from the IdP adapter, resource owner credentials or IdP connection (the inbound assertions for Service Providers) can also be used to map into OAuth access tokens (see “[Access Token Mapping](#)” on page 204).

Data stores may also be used here to retrieve any required user attributes.

Runtime Processing

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 169) but may be restricted on a client-by-client basis (see “[Client Management](#)” on page 190).

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

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 93).

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 184).

A *UserInfo* endpoint needed for the client to receive additional attributes with the access token is available as a standard OAuth-protected endpoint (see “[OpenID Connect Metadata Endpoint](#)” on page 590).

To configure policies, first enable the protocol in **Server Settings** (see “[Choosing Roles and Protocols](#)” on page 112). 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 169).

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 167.

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 and federation hub deployments PingFederate provides plug-in *authentication selectors*, which enable dynamic selection of authentication sources (IdP adapters, or IdP connections for federation hub use cases) based on administrator-specified criteria (see “[Bundled Authentication Selectors](#)” on page 16).

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 533)
- 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 235). 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 543 and “[Configuring the HTML Form IdP Adapter](#)” on page 550.)

Bundled Authentication Selectors

For IdP and federation hub deployments, PingFederate also includes global (cross-connection) authentication 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 authentication-selection criteria evaluation may be used to select subsequent selectors, which allows handling of complex hierarchical access-policy decisions (see “[Mapping Selector Results to Authentication Sources](#)” on page 262).

CIDR Authentication Selector – Provides a means of choosing an authentication source 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 determine, for example, whether an SSO request originates inside or outside the corporate firewall and use different authentication integration accordingly (see [“Configuring the CIDR Authentication Selector”](#) on page 257).

Cluster Node Authentication Selector – Provides a means of choosing an authentication source 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 Authentication Selector”](#) on page 257).

Connection Set Authentication Selector – Provides a means of selecting an authentication source 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 authentication selection on an individual connection basis (see [“Configuring the Connection Set Authentication Selector”](#) on page 258).



Note: Authentication selectors rely on HTTP Request/POST data—ensure that standard security measures are in place when using these selectors.

HTTP Header Authentication Selector – Provides a means of choosing an authentication source 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 Authentication Selector”](#) on page 259).

OAuth Scope Authentication Selector – Provides a means of choosing an authentication source 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 Authentication Selector”](#) on page 260).

Requested AuthN Context Authentication Selector – Provides a means of picking an authentication source 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 authentication sources 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 Authentication Selector”](#) on page 259).

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). Additional

authentication selectors may also be added to the download site periodically; contact your Ping Identity sales representative if you are looking for specific authentication-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)
- Access Token Management instances (see [“Access Token Management”](#) on page 175)
- Password Credential Validators (see [“Validating Password Credentials”](#) on page 235)
- Identity Store Provisioners (see [“Configuring Identity Store Provisioners”](#) on page 368)

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 127.

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 571). 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 354). 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 15 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 613)

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 362 or [“Configuring Token Generators”](#) on page 509). 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 285).



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 487). (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 487).

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 15 and [“Configuring IdP Adapters”](#) on page 250).

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 362.)

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 Target Session Mapping”](#) on page 393).

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 475).

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 509). 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 515).

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 122).

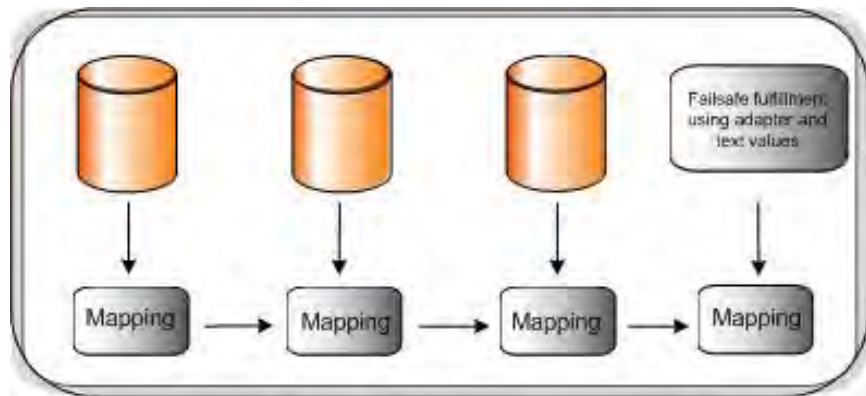
Alternatively, you can use the PingFederate Custom Source SDK to create your own driver for non-JDBC/LDAP data stores—including, for example, flat files or SOAP-connected databases (see the PingFederate SDK Developer's Guide).

Data stores can be used across multiple connections.

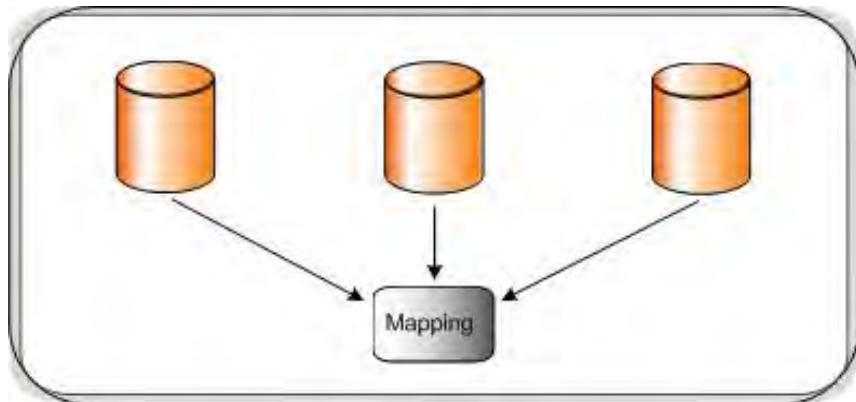
Multiple Data Source Attribute Mapping

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 294).



- 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 294).



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 50). 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 122).
- Retrieval of attributes from an IdP adapter or token processor (see [“Setting Pseudonym Values and Masking”](#) on page 254 and [“Setting Attribute Masking”](#) on page 482).
- SP-server processing of incoming attributes based on the SSO [attribute contract](#), (see [“Defining an Attribute Contract”](#) on page 392).

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 19).

- SP-server processing of incoming attributes in response to an Attribute Request under XASP (see [“Defining Security Policy”](#) on page 427).

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 50).

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 disallowing 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 307). The option is available for any IdP, SP, or OAuth configuration, or for direct SSO adapter-to-adapter or STS token-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 613).



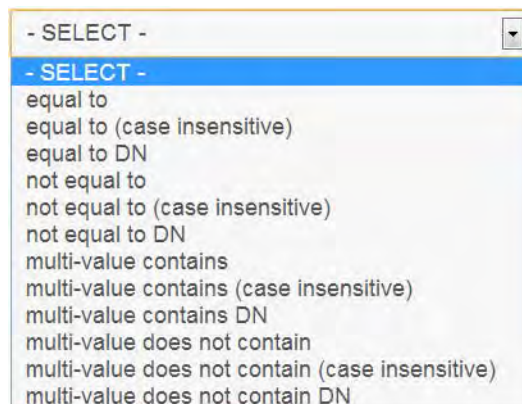
Tip: When more than one condition is configured, all are evaluated and all the conditions must be met at runtime (evaluated as “true”) for authorization to succeed and processing to continue. In cases where you might need “or” conditions or layered evaluations, you can create one or more OGNL expressions using an Advanced Criteria feature. For more information, see “[Using Attribute Mapping Expressions](#)” on page 613 and “[Expression Examples](#)” on page 616.



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 over certificate generation and authentication verification (see [“Security Management”](#) on page 221).

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 227).



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 230).

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 230). 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. By default, PingFederate also prompts for the Issuer DN of the certificates to mitigate potential man-in-the-middle attacks as well as to provide a means to isolate certificates used by different connections.

In addition, when the anchored trust model is chosen, 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), the Issuer DN (if specified) matches one of the issuers in the chain, 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 225).



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](http://www.oracle.com/technetwork/java/archive-139210.html) (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 227).

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 356 or [“Configuring IdP Auto-Connect”](#) on page 464.

Providing Metadata

You enable Auto-Connect as part of Server Settings from the Main Menu (see [“Choosing Roles and Protocols”](#) on page 112). 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 121).

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:

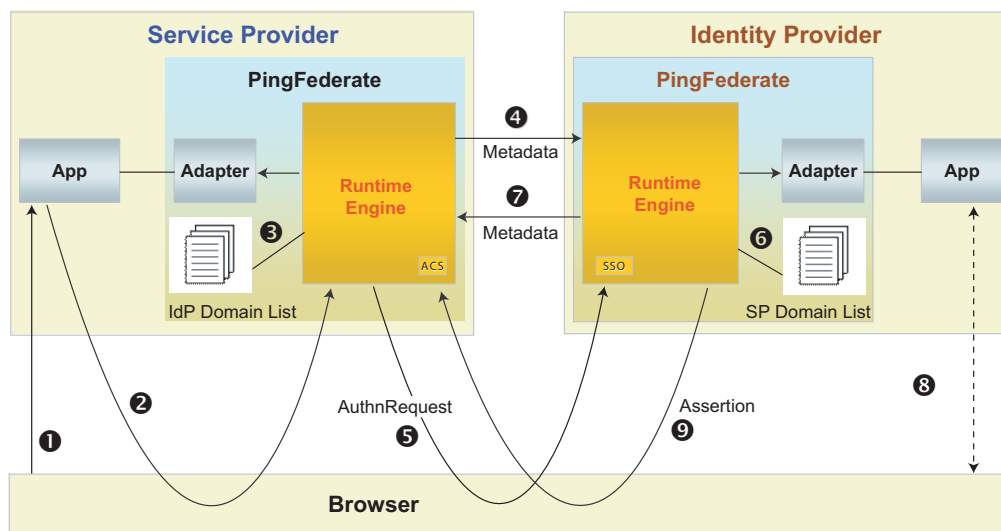


Figure 1: Auto-Connect Processing Flow

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:

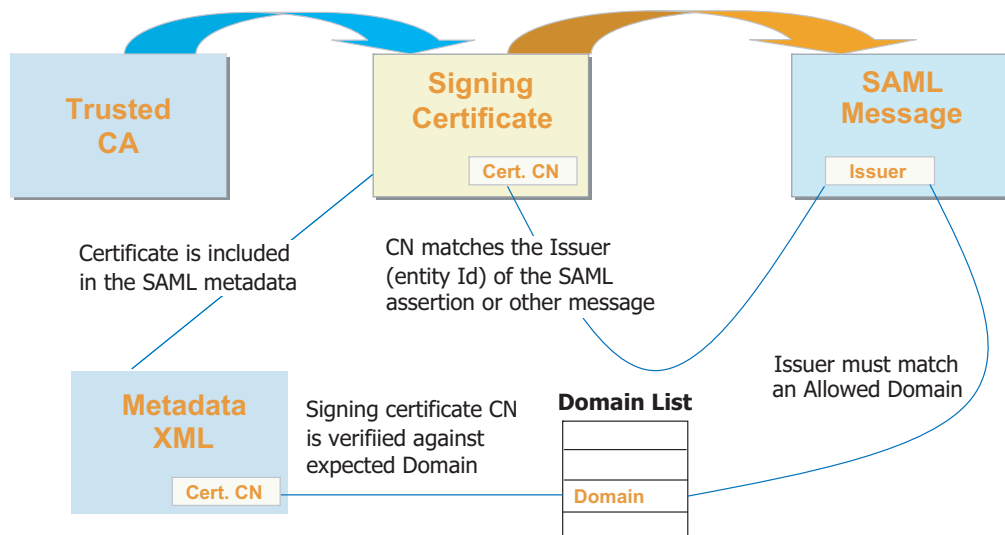


Figure 2: Auto-Connect Security Model

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 and groups 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 273). 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).

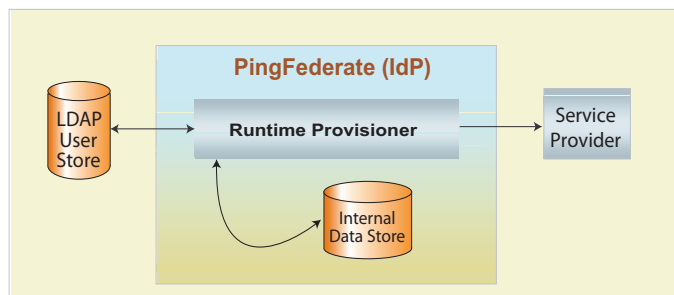


Figure 3: Outbound Provisioning Processing

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 343). 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 127).

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 119).

For more information, see [“Configuring Outbound Provisioning”](#) on page 339.

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 and group records in Microsoft AD data stores or custom user stores via the Identity Store Provisioners. 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 438. To integrate Inbound Provisioning with custom user stores, see [“Configuring Identity Store Provisioners”](#) on page 368. For application-development information about using PingFederate endpoints for SCIM provisioning, see [“SCIM Inbound Provisioning Endpoints”](#) on page 572.

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 382. For configuration information, see [“Using Just-in-Time Provisioning”](#) on page 427.

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 32).

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 40).

With PingFederate, you define a unique ID for each supported protocol (see [“Specifying Federation Information”](#) on page 114). 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 79).

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 15 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 50.)

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 18).

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 21.)

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 32.

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 63 and [“System-Services Endpoints”](#) on page 576 for more information).

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:

	Partner's federation ID	Your virtual server ID
IdP connection #1	eu.idp.local	idp-eu.sp.tld
IdP connection #2	na.idp.local	idp-na.sp.tld

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 276) or as an SP configuring an IdP connection (see [“General Connection Information”](#) on page 383) 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 563 as an IdP or [“SP Endpoints”](#) on page 567 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 Authentication Source to Certain Virtual Server IDs”](#) on page 291 and [“Restricting a Target Session to Certain Virtual Server IDs”](#) on page 396).



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 491 or [“Restricting a Token Generator to certain Virtual Server IDs”](#) on page 517).



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 307 as an IdP or [“Identifying Issuance Criteria \(Optional\)”](#) on page 408 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 20).
- **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.

Federation Hub

PingFederate can be configured as a federation hub to:

- Bridge partners using different federation protocols to circumvent partner or application limitations.
- Multiplex a connection for multiple partners to reduce costs and expand use cases.

As a federation hub, PingFederate can bridge browser-based SSO between identity providers and service providers. It stands in the middle of the SSO flow, acting as the SP for the identity providers and as the IdP for the service providers. The four use cases are:

- [Bridging an IdP to an SP](#)
- [Bridging an IdP to multiple SPs](#)
- [Bridging multiple IdPs to an SP](#)
- [Bridging multiple IdPs to multiple SPs](#)

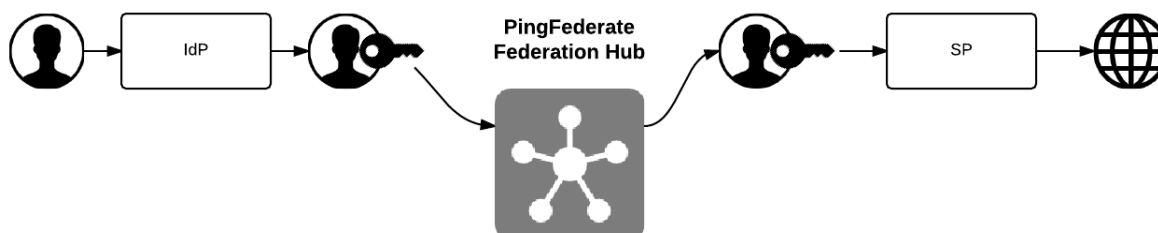
PingFederate also supports protocol translation among SAML 1.0, 1.1, 2.0 and WS-Federation. For SAML based connections, this also means it is possible to bridge between various bindings between identity providers and service providers.



Note: The federation hub capability can be deployed alongside with other OAuth, IdP and/or SP connections to your partners. This flexibility helps in streamlining your federation infrastructure and reducing operating costs.

Bridging an IdP to an SP

In this use case, PingFederate is bridging SSO transactions between an identity provider and a service provider. For example, you may have a legacy IdP system that is only capable of sending SAML 1.1 assertions via POST. Your service provider however requires SAML 2.0 assertions via the artifact binding. With federation hub, you can configure PingFederate to consume inbound SAML 1.1 assertions (by POST), translate them to SAML 2.0 assertions, and send them via the artifact binding to the service provider.

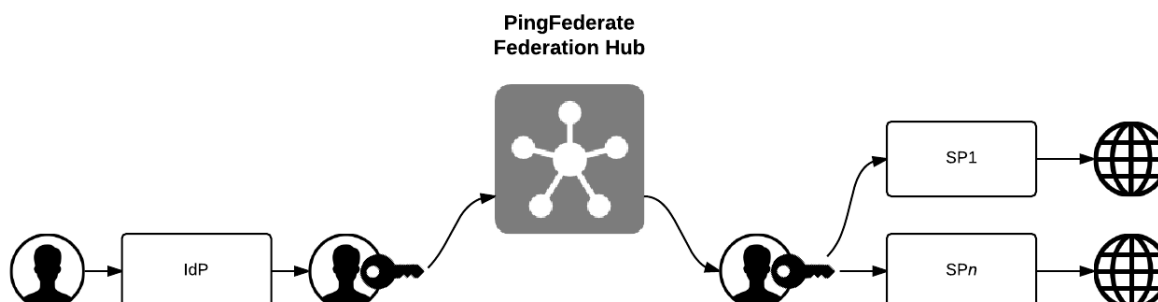


To address this use case:

1. Enable both the IdP and the SP roles with the applicable protocols (see [“Choosing Roles and Protocols”](#) on page 112).
2. Create a contract to bridge the attributes between the identity provider and the service provider (see [“Connection Mapping Contracts”](#) on page 162).
3. Create an IdP connection between the identity provider and PingFederate (the federation hub as the SP). In the Target Session Mapping screen, add the connection mapping contract into the connection (see [“Configuring Target Session Mapping”](#) on page 393).
4. Create an SP connection between PingFederate (the federation hub as the IdP) and the service provider. In the Authentication Source Mapping screen, add the connection mapping contract into the connection (see [“Authentication Source Mapping”](#) on page 287).
5. Work with the identity provider to connect to PingFederate (the federation hub) as the SP.
6. Work with the service provider to connect to PingFederate (the federation hub) as the IdP.

Bridging an IdP to multiple SPs

In this use case, PingFederate is bridging SSO transactions between an identity provider and multiple service providers. For example, your company wants to route federation requests from a recently acquired subsidiary through its federation infrastructure. With PingFederate, you can multiplex one IdP connection to multiple SP connections to the desired service providers. The federation hub consumes assertions from the subsidiary and creates new assertions to the respective service providers.

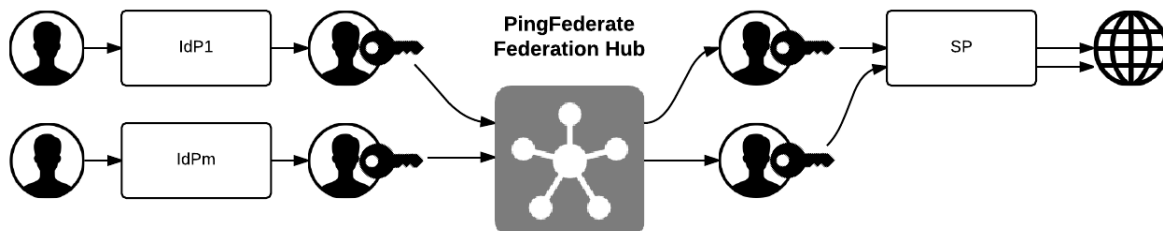


To address this use case:

1. Enable both the IdP and the SP roles with the applicable protocols (see [“Choosing Roles and Protocols”](#) on page 112).
2. For each service provider, create a contract to the identity provider (see [“Connection Mapping Contracts”](#) on page 162). Multiple contracts are likely required, because each service provider may require a unique set of attributes.
3. Create an IdP connection between the identity provider and PingFederate (the federation hub as the SP). In the Target Session Mapping screen, add the connection mapping contracts into the connection (see [“Configuring Target Session Mapping”](#) on page 393).
4. For each service provider, create an SP connection between PingFederate (the federation hub as the IdP) and the service provider. In the Authentication Source Mapping screen, add the corresponding connection mapping contract into the connection (see [“Authentication Source Mapping”](#) on page 287).
5. For each service provider supporting the SAML IdP-initiated SSO profile, map the expected Target Resource URLs to the corresponding SP connection (see [“Configuring Target URL Mapping”](#) on page 366).
6. Work with the identity provider to connect to PingFederate (the federation hub as the SP).
7. Work with each service provider to connect to PingFederate (the federation hub as the IdP).

Bridging multiple IdPs to an SP

In this use case, PingFederate is bridging SSO transactions between multiple identity providers and a service provider. For example, you are tasked to provide federated access to resources on Microsoft® SharePoint® for various business partners. With PingFederate, you can multiplex one SP connection (to SharePoint) to multiple IdP connections for all your business partners. The federation hub can also, as needed, translate SAML assertions from the business partners to WS-Federation security tokens and send them over to SharePoint.



To address this use case:

1. Enable both the IdP and the SP roles with the applicable protocols (see [“Choosing Roles and Protocols”](#) on page 112).
2. Create a contract to bridge the attributes between the identity providers and the service provider (see [“Connection Mapping Contracts”](#) on page 162).

You likely need only one contract unless the service provider requires a different set of attributes from each identity provider.

3. For each identity provider, create an IdP connection between the identity provider and PingFederate (the federation hub as the SP). In the Target Session Mapping screen, add the corresponding connection mapping contract into the connection (see [“Configuring Target Session Mapping”](#) on page 393).
4. Configure an authentication selector to map each identity provider to the corresponding IdP connection (see [“Configuring Authentication Selectors”](#) on page 255).
5. Create an SP connection between PingFederate (the federation hub as the IdP) and the service provider. In the Authentication Source Mapping screen, add the connection mapping contract into the connection (see [“Authentication Source Mapping”](#) on page 287).



Important: PingFederate includes the Entity ID of the original identity provider (*Authenticating Authority*) in SAML 2.0 assertions so that the service provider can determine the original issuer of the assertions. This is especially important when bridging multiple identity providers to one service provider—the service provider should take the information about the original issuer into consideration before granting access to protected resources.

For SAML 1.x assertions and WS-Federation security tokens, you can add an attribute to the Attribute Contract (see [“Creating an Attribute Contract”](#) on page 285) and map *Authenticating Authority* into its value (see [“Attribute Contract Fulfillment”](#) on page 304).

For information about *Authenticating Authority*, see section 2.7.2.2 Element <AuthnContext> in SAML 2.0 specification (docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf).

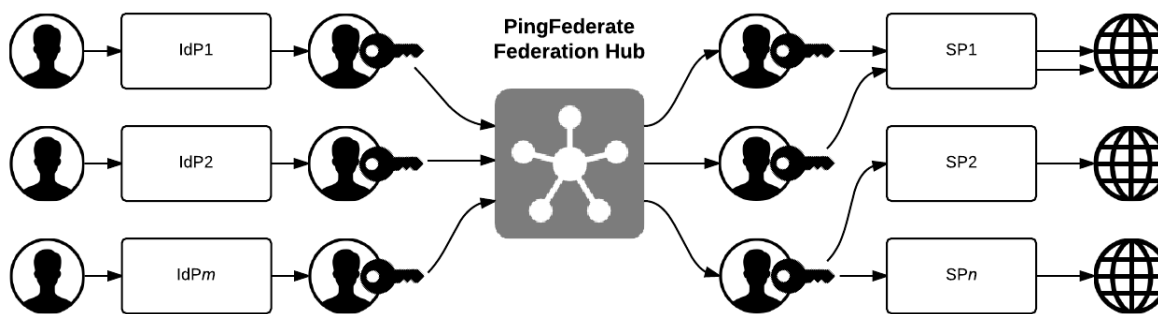


Note: If the service provider does not take action based on *Authenticating Authority*, depending on the attributes from the identity providers, you may use Issuance Criteria in the IdP connection to protect against user impersonation between IdPs.

6. Work with each identity provider to connect to PingFederate (the federation hub as the SP).
7. Work with the service provider to connect to PingFederate (the federation hub as the IdP).

Bridging multiple IdPs to multiple SPs

This use case is a combination of [“Bridging an IdP to multiple SPs”](#) on page 43 and [“Bridging multiple IdPs to an SP”](#) on page 44.



To address this use case:

1. Enable both the IdP and the SP roles with the applicable protocols (see [“Choosing Roles and Protocols”](#) on page 112).
2. Create multiple contracts to bridge the attributes between the identity providers and the service providers (see [“Connection Mapping Contracts”](#) on page 162).
3. For each identity provider, create an IdP connection between the identity provider and PingFederate (the federation hub as the SP). In the Target Session Mapping screen, add the applicable connection mapping contract(s) into the connection (see [“Configuring Target Session Mapping”](#) on page 393).
4. Configure an authentication selector to map each identity provider to the corresponding IdP connection (see [“Configuring Authentication Selectors”](#) on page 255).
5. For each service provider, create an SP connection between PingFederate (the federation hub as the IdP) and the service provider. In the Authentication Source Mapping screen, add the corresponding connection mapping contract into the connection (see [“Authentication Source Mapping”](#) on page 287).



Important: PingFederate includes the Entity ID of the original identity provider (*Authenticating Authority*) in SAML 2.0 assertions so that the service provider can determine the original issuer of the assertions. This is especially important when bridging multiple identity providers to one service provider—the service provider should take the information about the original issuer into consideration before granting access to protected resources.

For SAML 1.x assertions and WS-Federation security tokens, you can add an attribute to the Attribute Contract (see [“Creating an Attribute Contract”](#) on page 285) and map *Authenticating Authority* into its value (see [“Attribute Contract Fulfillment”](#) on page 304).

For information about *Authenticating Authority*, see section 2.7.2.2 Element <AuthnContext> in SAML 2.0 specification (docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf).



Note: If the service provider does not take action based on *Authenticating Authority*, depending on the attributes from the identity providers, you may use Issuance Criteria in the IdP connection to protect against user impersonation between IdPs.

6. For each service provider supporting the SAML IdP-initiated SSO profile, map the expected Target Resource URLs to the corresponding SP connection (see [“Configuring Target URL Mapping”](#) on page 366).
7. Work with each identity provider to connect to the federation hub (as the SP).
8. Work with each service provider to connect to the federation hub (as the IdP).

Federation Hub and Virtual Server IDs

PingFederate uses two connections to bridge an identity provider to a service provider:

- An IdP connection where end users authenticate and PingFederate (the federation hub) is the SP
- An SP connection to the target application where PingFederate (the federation hub) is the IdP

Generally speaking, PingFederate consumes assertions from the identity provider through the IdP connection and generates new assertions to the service provider via the SP connection.

If the SP connection does not use a virtual server ID (see [“General Information”](#) on page 276), the issuer of the assertions (to the service provider) is the ID defined for the protocol between PingFederate (the federation hub as the IdP) and the service provider (see [“Specifying Federation Information”](#) on page 114).

If the SP connection uses multiple virtual server IDs (see [“Connecting to a Partner in One Connection”](#) on page 39), for SP-initiated SSO, if the service provider sends AuthnRequest messages to the virtual server ID specific endpoint (see step 3 under [“To export connection metadata”](#) in [“Exporting Metadata”](#) on page 63), PingFederate retains this information automatically. When the identity provider returns the corresponding assertions to PingFederate (the federation hub as the SP), PingFederate retrieves the preserved information and uses that specific virtual server ID as the issuer in the assertions it sends to the service provider. For IdP-initiated SSO, the issuer of the assertions (to the service provider) is the default Virtual Server ID.

System Administration

This chapter describes general administrative functions for PingFederate, including:

- [“Starting and Stopping PingFederate”](#) on page 50
- [“Managing Log Files”](#) on page 50
- [“Exporting Metadata”](#) on page 63
- [“Signing XML Files”](#) on page 67
- [“Using the Configuration Archive Utility”](#) on page 68
- [“Account Management”](#) on page 71
- [“Alternative Console Authentication”](#) on page 74
- [“Managing Email Configuration”](#) on page 77
- [“Using Virtual Host Names”](#) on page 79
- [“Changing Configuration Parameters”](#) on page 80
- [“Installing a New License Key”](#) on page 83
- [“Automating Configuration Migration”](#) on page 84
- [“Outbound Provisioning CLI”](#) on page 89
- [“Customizing User-Facing Screens”](#) on page 93
- [“Configuring Password Policy”](#) on page 100
- [“Extending the Lifetime of the PF Cookie”](#) on page 100
- [“Adding Custom HTTP Response Headers”](#) on page 101
- [“Customizing the Favicon for Application and Protocol Endpoints”](#) on page 101



Note: The information in this chapter is presented from the viewpoint of an administrative user with “Admin” permissions (see [“Account Management”](#) on page 71).

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]s:[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]s:[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 52).
- `admin-api.log` — Records all actions performed by administrative-api users (see “[Administrator API Audit Logging](#)” on page 53).
- `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 53).

- `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 54).
- `provisioner-audit.log` — Records Outbound Provisioning events, intended for security-audit purpose (see [“Outbound Provisioning Audit Logging”](#) on page 56).
- `server.log` — Records all PingFederate runtime and administrative server operational activity (see [“Server Logging”](#) on page 57). PingFederate provides a utility for filtering server log entries (see [“Using the Server Log Filter”](#) on page 57).
- `provisioner.log` — Records only provisioning activity.
(For more information, see [“Outbound Provisioning for IdPs”](#) on page 35.)
- `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 25).

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 80).



Note: The server, provisioner, and audit logs may be output to alternate formats, including database tables (see [“Writing Logs to Other Formats”](#) on page 59).

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”](#)
- [“Administrator API Audit Logging”](#) on page 53
- [“Runtime Transaction Logging”](#) on page 53
- [“Security Audit Logging”](#) on page 54
- [“Outbound Provisioning Audit Logging”](#) on page 56
- [“Server Logging”](#) on page 57
- [“Writing Logs to Other Formats”](#) on page 59

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

- SP Adapter created, modified, or deleted
- IdP Adapter created, modified, or deleted
- Server settings management
- Metadata export
- Configuration archive
- 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 606). 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 2: Transaction Logging Modes

Mode	Description
None	No transaction logging.
Standard	(Default) Logs summary information for each transaction message, including: <ul style="list-style-type: none">• 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: <ul style="list-style-type: none">• 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-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 56).

Audit-log elements may be output to different formats, including databases. For information, see [“Writing Logs to Other Formats”](#) on page 59.

Table 3: Audit Log Configuration

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

Table 3: Audit Log Configuration (Continued)

Item*	Description
responsetime	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.
role	The partner's role in the transaction.
status	Transaction success or failure.
<i>stspluginid</i>	For STS transactions, the ID for the token-processor or token-generator instance.
subject	The subject of the transaction.
<i>trackingid</i>	The tracking ID used for debugging purposes in the server log (see the last "Note" on page 51 in the section "Managing Log Files").
validatorid	The ID of the Password Credential Validator used for OAuth Resource Owner Grant transactions (see "Resource-Owner Credentials Mapping" on page 195).

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

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 107)..

Elements from the provisioner audit log may be output to different formats, including databases. For information, see "Writing Logs to Other Formats" on page 59.

Table 4: Provisioner Audit Log Configuration

Item	Description
%d	Transaction time.
cycle_id	The unique ID for each provisioning cycle.
channel_id	The unique ID for Provisioning Channel between source and target.

Table 4: Provisioner Audit Log Configuration (Continued)

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

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 59).

To facilitate troubleshooting, administrators can use a filter utility to aggregate related events (see [“Using the Server Log Filter”](#) on page 57).

Elements of this log are described in the following table.

Table 5: Server Log Components

Item	Description
%d	Event date and time.
%X{trackingid}	The tracking ID for runtime events, used for debugging purposes (see the last “Note” on page 51 in the section “Managing Log Files”).
%p	Logging level.
%c	The Java class issuing the status or error message, when applicable.
%X{connectionid}	The partner ID associated with a runtime event.
%X{subject}	The SAML subject of the transaction, when applicable.
%m	Status or error message.

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 6: Server Log Filter Command Options

Table 7: Server Log Filter Command Options

Command Option	Description
- entityid <entity-id> - subject <subject>	These two commands must be used together and return a list of transactions for the specified federation partner's entity ID and transaction subject.
- trackingid <tracking_id>	This command returns a list of transactions with the same tracking ID.
- sessionxrefid <session_xref_id>	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. The value of <session_xref_id> may be the value of any of the following transaction tags in the target server log(s): <ul style="list-style-type: none"> • Artifact • Session Index • Assertion ID

Table 8: Server Log Filter Parameters

Table 9: Server Log Filter Parameters*

Parameter*	Description
- logsdire <log-files-directory>	Full or relative path to source directory for the log(s). Default: all server.log files in pf.log.dir The installed location is: <pf_install>/pingfederate/log (For information about changing the default location, see "Changing Configuration Parameters" on page 80.)

Table 9: Server Log Filter Parameters* (Continued)

Parameter*	Description
- outputfile <output-file>	Output path and file for the returned list. Default: <code>pf.log.dir/logfilter_output.log</code> The installed location is: <code><pf_install>/pingfederate/log</code> (For information about changing the default location, see “Changing Configuration Parameters” on page 80.)
- 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.

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 61).

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 62).

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.

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 3](#) on page 55 for the audit log and [Table 4](#) on page 56 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`

- 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 376 or “Managing SP Connections” on page 267).



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 65.
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 39).

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 66.

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 65.

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 65
 - “[Including a Signing Key](#)” on page 66
 - “[Signing Metadata](#)” on page 66
 - “[XML Encryption Certificates](#)” on page 67
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 66.

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 80.)

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 21.

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:

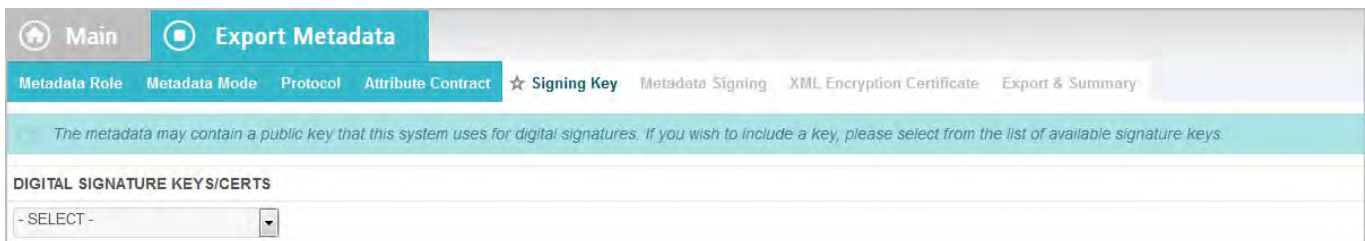
1. Click **Edit** and make your change.
2. Click **Update**.

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 227.




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 227).

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 is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection 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 383.
- As an IdP configuring an SP connection, see [“Importing Metadata”](#) on page 275.



- ▶ XML file signing is available from the Main Menu under Administrative Functions.

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.



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.

3. On the Export & Summary screen, click the **Export** button to save the signed file.



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

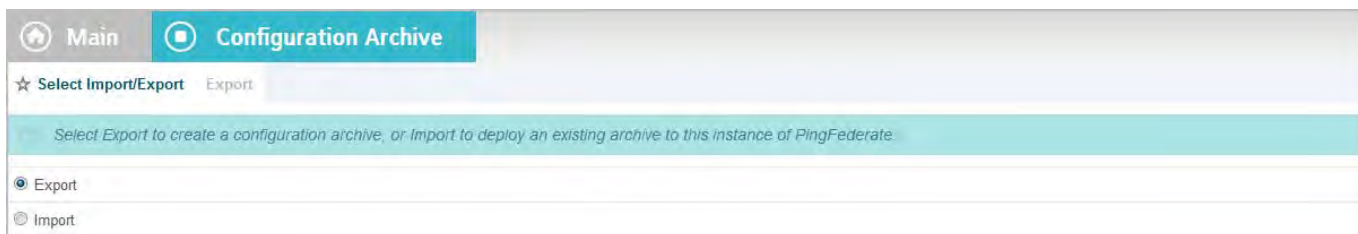
Using the Configuration Archive Utility

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`. Configuration archives can be used as backups.

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 84).





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 93), 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 104).



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 74).

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 10: PingFederate User Access Control

Role Assignment	Access Privileges
User Admin	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.)
Admin	Configure partner connections and most system settings (except user management and local key/certificate handling).
Crypto Admin	Manage local keys and certificates.
Auditor	View-only permissions for all administrative functions.

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 80).

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 74).

USERNAME	USER ADMIN	ADMIN	CRYPTO ADMIN	ACTION	
Administrator	<input type="radio"/> Auditor <input checked="" type="radio"/> Admin	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Deactivate / Change Password

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 74).

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 73).



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 77).
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 74).

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 100.
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.

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 71).

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 100.



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 71, 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:

```
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 71.



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 71.)

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 222).

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 71.)

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 71 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 50).

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 71 and [“Configuring Runtime Notifications”](#) on page 105).

Field Descriptions

Field	Description
"From" Address	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.
Email Server	The IP address or hostname of your email server.
SMTP Port	The SMTP port on your email server (default: 25).
SSL SMTP Port	The secure SMTP port on your email server (default: 465). This field is not active unless Use SSL is enabled below.
Connection Timeout	The amount of time in seconds that PingFederate will wait before it times out connecting to the SMTP server.
Use SSL	(Optional) Requires the use of the SMTP secure channel.
Use TLS	(Optional) Requires the use of the secure transport layer.
Enable SMTP Debugging Messages	(Optional) Turns on detailed error messages for the PingFederate server log to help troubleshoot any problems,
Username	(Optional) Authorized email username.

Field	Description
Password	(Optional) User password.
Confirm Password	Re-entered password, when a Password is used.
Test Address	(Optional) An email address to use in conjunction with testing the connection.

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 105 or [“Account Management”](#) on page 71).

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 38). Depending on your needs, however, you can configure virtual server IDs and virtual hosts in the same installation of PingFederate.

★ Virtual Host Names

Optionally, you can define a list of alternate domain names at which this server may receive XML protocol messages. This option allows more flexibility for validating protocol elements such as `<Recipient>` and `<Destination>` in inbound messages.

HOST DOMAIN NAME	ACTION
<input type="text"/>	<input type="button" value="Add"/>

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 11: PingFederate Configuration Properties

Table 12: PingFederate Configuration Properties

Property	Description
<code>pf.admin.https.port</code>	Defines the port on which the PingFederate administrative console runs. Default is 9999.
<code>pf.console.bind.address</code>	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.
<code>pf.console.title</code>	Defines the browser window or tab title for the administrative console, used to make separate instances identifiable.
<code>pf.console.session.timeout</code>	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.
<code>pf.console.login.mode</code>	Indicates whether more than one Admin user may access the administrative console at one time (see "Setting Administration Options" on page 104). Values: <code>Single</code> <code>Multiple</code> . Default is <code>Multiple</code> .
<code>pf.console.authentication</code>	Indicates whether administrators log on to PingFederate using credentials managed internally, by PingFederate, or externally (see "Alternative Console Authentication" on page 74).

Table 12: PingFederate Configuration Properties (Continued)

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

Table 12: PingFederate Configuration Properties (Continued)

Property	Description
pf.monitor.bind.address	Defines the IP address over which an SNMP agent and JMX communicate with PingFederate (see “Configuring Runtime Reporting” on page 106). Use for deployments where multiple network interfaces are installed on the machine running PingFederate.
pf.engine.prefer_ipv4	Defines the protocol to be used by PingFederate. <code>True</code> (the default) enables use of <code>ipv_4</code> only. <code>False</code> enables use of both <code>ipv_4</code> and <code>ipv_6</code> .
pf.runtime.context.path	Allows customization of the server path for PingFederate endpoints (see “Application Endpoints” on page 563). 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 114).
pf.log.dir	Network path to the output location of log files. The default is: <code><pf_install>/pingfederate/log</code>
pf.hsm.mode	Enables or disables (the default) a FIPS-compliance Hardware Security Module (see Appendix A in <i>Getting Started</i>).
pf.provisioner.mode	Enables or disables (the default) Outbound Provisioning (see “Outbound Provisioning for IdPs” on page 35). Also used to enable provisioning failover (see the PingFederate Server Clustering Guide).

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 105).

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](http://www.pingidentity.com/support/licensing) (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 50).
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 592).



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 233).



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

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:

```
{ "keys" : [ { "kty" : "oct", "kid" : "j0PUEdAb95", "k" : "AGi8Lg_ewdl-_30Cx83kDMQE9oNlhgJSa_Pc4I8JTU8" } ] }
```

3. In your PingFederate installation on the target sever, 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:

```
{ "keys" : [ { "kty" : "oct", "kid" : "wER9zEpaPe", "k" : "i0HQr9Jm sqjAX4o_BQU1qGJzoLQI-nmwp8u3GyHzTB8" }, { "kty" : "oct", "kid" : "j0PUEdAb95", "k" : "AGi8Lg_ewdl-_30Cx83kDMQE9oNlhgJSa_Pc4I8JTU8" } ] }
```

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 68). 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)

```
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)

```
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:

```
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 233). 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 data-store configurations), be sure to adhere closely to the instructions provided in the following procedure.

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 233).
2. Determine which component configurations need to be copied, including plug-ins.

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 221).

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 REAME.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 a command-line interface (CLI) to help manage automated outbound provisioning at IdP sites (see [“Outbound Provisioning for IdPs”](#) on page 35). 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 339).

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 119).

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 13: Outbound Provisioning CLI Global Options

Command Argument	Description
<code>--help</code>	Describes the available options. The help is also displayed if the command is run with no arguments.

Table 13: Outbound Provisioning CLI Global Options (Continued)

Command Argument	Description
--show-channels	Lists all channels in a table format, showing for each: <ul style="list-style-type: none"> • ID - A numeric channel ID (channel-specific commands need this ID) • Name - The channel name • Connection ID • Status (active/inactive) - Both the connection and the channel status are shown (see "Channel Activation and Summary" on page 352) • User count/dirty-user-record count (e.g.: 5000/12 means 5000 users and 12 dirty records) • Source (as LDAP URL) • Target code
--show-nodes	Shows all the provisioning-server nodes with their status and the last timestamp (applies only to a failover configuration—see the <i>PingFederate Server Clustering Guide</i>).
--force-node-backup Use with node number: -n <node ID>	Sets the provisioner mode to <code>FAILOVER</code> for the associated PingFederate server node (see the <i>Server Clustering Guide</i>).

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 14: Outbound Provisioning CLI Channel-Specific Options

Command Argument	Description
--reset-group-timestamp	<p>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.</p> <p>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.</p> <p>Important: 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.</p>
--reset-attribute-sync	<p>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.</p> <p>Important: 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.</p>
--reset-values-hash	<p>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.)</p> <p>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 <i>not</i> updated.</p>
--reset-all	<p>Equivalent to using all three of the arguments above.</p> <p>Important: 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.</p>
--show-dirty-records	Lists all users or groups that have not been provisioned or updated at the service-provider site.
--show-dirty-group-records	List groups that have not been provisioned or updated at the service-provider site.
--show-dirty-user-records	List all users that have not been provisioned or updated at the service-provider site.

Table 14: Outbound Provisioning CLI Channel-Specific Options (Continued)

Command Argument	Description
--show-group --show-user Use with: -u <provider name> Or: -g <LDAP GUID>	<p>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.</p> <p>Note: You can obtain user or group names and GUIDs for dirty records, as needed, using any of the <code>--show-dirty-*</code> options (described above). The LDAP GUID, if used and if it is binary, should be entered in hexadecimal format (as shown in log files).</p> <p>Examples:</p> <pre> provmgr.sh --show-user -u john@example.com provmgr.sh --show-user -g ffd448643f812b43a0bee2504173f0 </pre>
--clear-dirty-records	Clears the dirty flag on all records.
--clear-dirty-group-records	Clears the dirty flag on all group records.
--clear-dirty-user-records	Clears the dirty flag on all user records.
--delete-dirty-records	Removes all dirty records from the internal store.
--delete-dirty-group-records	Removes all dirty group records from the internal store.
--delete-dirty-user-records	Removes all dirty user records from the internal store.
--delete-all --delete-all-users	<p>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.</p> <p>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 119).</p> <p>Important: These options should be needed rarely if ever. If needed, you may wish to schedule the operation when other network activity is low.</p>
--show-target	Displays the target configuration.
--show-source	Displays all source LDAP configuration parameters, including settings and location.

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.

Variable	Description
<code>\$escape</code>	A utility class that can be used to escape String variables inserted into the template, for example, <code>\$escape.escape(\$client.name)</code> where <code>\$client.name</code> is a second variable available for the page.
<code>\$HttpServletRequest</code>	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).
<code>\$HttpServletResponse</code>	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).
<code>\$locale</code>	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.
<code>\$PingFedBaseUrl</code>	The PingFederate base URL (see "Specifying Federation Information" on page 114).
<code>\$templateMessages</code>	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><pf_install>/pingfederate/sdk/doc</code> . For more information about localization, see "Localization" on page 98.
<code>\$TrackingId</code>	The user's session tracking ID (see the last "Note" on page 51 in the section "Managing Log Files").

For information about Velocity, please refer to the Velocity project documentation on the Apache Web site:

<http://velocity.apache.org/engine/releases/velocity-1.4>

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 563). 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 264; for the SP, see "Configuring Default URLs" on page 372).

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 98).

IdP User-Facing Pages

Table 15: IdP User-Facing Pages

Page Title <i>and template file name</i>	Purpose	Type	Action
Change Password <i>html.form.change.password.template.html</i>	Allows a user to change his or her password via the HTML Form Adapter (see "Configuring the HTML Form IdP Adapter" on page 550).	Normal	User input required
Change Password Message <i>html.form.message.template.html</i>	Displayed when a user has successfully changed his or her password (see "Configuring the HTML Form IdP Adapter" on page 550).	Normal	User input required
Error - Single Logout <i>idp.slo.error.page.template.html</i>	Displayed when an SLO request fails and no other SLO error landing page is specified.	Error	User should close browser
Error - Single Sign-On <i>idp.sso.error.page.template.html</i>	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.	Error	Consult log and Web developer
IdP Logout <i>idp.logout.success.page.template.html</i>	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 543 and "Configuring the HTML Form IdP Adapter" on page 550).	Normal	None

Table 15: IdP User-Facing Pages (Continued)

Page Title <i>and template file name</i>	Purpose	Type	Action
IdP Logout Confirmation <i>idp.slo.confirm.page.template.html</i>	Displayed to prompt the user to confirm Single Logout (SLO). For more information, see “Configuring a Default URL and Error Message” on page 264).	Normal	User input required
Password Management System Message <i>html.form.message.template.html</i>	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 550).	Normal	User input required
Please Select Authentication System <i>sourceid-choose-idp-adapter-form-template.html</i>	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 <i>html.form.login.challenge.template.html</i>	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 550).	Normal	User input required
Sign On <i>html.form.login.template.html</i>	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 550).	Normal	User input required
Signed Out <i>sourceid-wsfed-idp-signout-cleanup-template.html</i>	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 <code>/idp/startSLO.ping</code> endpoint (see “Asynchronous Front-Channel Logout” on page 167).	Normal	None
Signing Out <i>sourceid-wsfed-idp-signout-cleanup-invisible-template.html</i>	WS-Federation and OpenID Connect Client IdP sign-out processing page. Note: No HTML is rendered in the browser.	Normal	None
Success - Single Logout <i>idp.slo.success.page.template.html</i>	Displayed when an SLO request succeeds but no other SLO landing page is specified.	Normal	None
User Consent <i>consent-form-template.html</i>	Displayed when a request requires a user’s consent for an SSO to an SP.	Normal	User input required

Table 15: IdP User-Facing Pages (Continued)

Page Title <i>and template file name</i>	Purpose	Type	Action
Working . . . <i>sourceid-wsfed-http-post-template.html</i>	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. Note: Normally not displayed if Javascript executes properly.	Normal	None

SP User-Facing Pages

Table 16: SP User-Facing Pages

Page Title <i>and template file name</i>	Purpose	Type	Action
Account Link Removed <i>TerminateAccountLinks.page.template.html</i>	Communicates a user's successful "defederation" operation.	Normal	None
Account Linking <i>LocalIdPasswordLookup.form.template.html</i>	Used to authenticate a user at the SP when an account link needs to be established.	Normal	None
Authentication Failed <i>sourceid-wsfed-idp-exception-template.html</i>	Displayed when an authentication challenge fails during WS-Federation processing.	Error	Consult log and Web developer
Error - Single Logout <i>sp.slo.error.page.template.html</i>	Displayed when an SLO request fails and no other SLO error landing page is specified.	Error	User should close the browser
Error - Single Sign-On <i>sp.sso.error.page.template.html</i>	Displayed when SP-initiated SSO fails (or IdP-initiated SSO fails on the SP side) and no other SSO error landing page is specified.	Error	Consult log and Web developer
Select Identity Provider <i>sourceid-saml2-idp-selection-template.html</i>	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.	Normal	User must make selection
Signed Out - Service Provider <i>sourceid-wsfed-sp-signout-cleanup-template.html</i>	Displays the user's sign-out status.	Normal	None
Success - Single Logout <i>sp.slo.success.page.template.html</i>	Displayed when an SLO request succeeds and no other SLO success landing page is specified.	Normal	None

Table 16: SP User-Facing Pages (Continued)

Page Title and template file name	Purpose	Type	Action
Single Sign-On Target Unspecified <i>sp.sso.success.page.template.html</i>	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 372).	Error	Consult Web developer, or specify default URL

Either IdP or SP User-Facing Pages

Table 17: Either IdP or SP User-Facing Pages

Page Title and template file name	Purpose	Type	Action
Error - Single Sign-On <i>generic.error.msg.page.template.html</i>	For an Auto-Connect SSO transaction, indicates a range of possible error conditions (see "Using Auto-Connect" on page 32): <ul style="list-style-type: none"> The requesting Auto-Connect partner is not found in the PingFederate server's list of allowed domains. The partner's metadata is not accessible. The server is not configured for Auto-Connect. General error, with error code. 	Error	Consult log, check configuration, or contact partner. If unresolved, contact Ping Identity support .
Error <i>general.error.page.template.html</i>	Indicates that an unknown error has occurred and provides a error reference number and (optionally) an error message.	Error	Consult log, contact Ping Identity support
Multiple SSO in Progress <i>speed.bump.template.html</i>	Displayed to a user when response is slow due to simultaneous SSO requests coming from multiple browser tabs.	Normal	None
Page Expired <i>state.not.found.error.page.template.html</i>	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.	Error	None
Sign On <i>AbstractPasswordIdpAuthnAdapter.form.template.html</i>	Challenges user for credentials when authentication can take place via HTTP Basic Authentication or an HTML form, depending on the operational mode.	Normal	User must sign on

Table 17: Either IdP or SP User-Facing Pages (Continued)

Page Title <i>and template file name</i>	Purpose	Type	Action
Submit Form <i>form.autopost.template.html</i>	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. Note: Normally not displayed if Javascript executes properly.	Normal	None

OAuth User-Facing Pages

Table 18: OAuth User-Facing Pages

Page Title <i>and template file name</i>	Purpose	Type	Action
Access Revocation <i>oauth.access.grants.page.template.html</i>	Provides a means for end-users (resource owners) to revoke persistent access grants.	Normal	None
Information Access Approval <i>oauth.approval.page.template.html</i>	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 169). Tip: The OAuth client configuration provides an option to bypass this approval page entirely, as needed for trusted clients (see “Configuring a Client” on page 191).	Normal	None

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 99).

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 `PingFederate` uses to parse the cookie: `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`.

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



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

To configure the complexity of the password policy:

1. Edit `<pf_install>/pingfederate/server/default/data/config-store/password-rules.xml`.
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.

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:

1. Edit `<pf_install>/pingfederate/server/default/data/config-store/session-cookie-config.xml`.
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.

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

1. On the console node, edit `<pf_install>/pingfederate/server/default/data/config-store/session-cookie-config.xml`.
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. For more information, see [“Configuring the HTML Form IdP Adapter”](#) on page 550.

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:

1. Edit `<pf_install>/pingfederate/server/default/data/config-store/response-header-admin-config.xml` or `response-header-runtime-config.xml`.
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.

Customizing the Favicon for Application and Protocol Endpoints

PingFederate provides a favorite icon (favicon) for its Application and Protocol Endpoints (see [“Application Endpoints”](#) on page 563, [“Viewing Protocol Endpoints”](#) on page 265 for IdP and [“Viewing SP Protocol Endpoints”](#) on page 374).

To change the favorite icon:

1. Replace the `favicon.ico` file in the `<pf_install>/pingfederate/server/default/conf/template/assets/images` folder.
2. Restart PingFederate.
3. Repeat these steps on each engine node if you have a clustered PingFederate environment.

System Settings

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 103
- [“Managing Data Stores”](#) on page 122
- [“Configuring IdP Discovery”](#) on page 139
- [“Configuring Redirect Validation”](#) on page 143



Note: The information in this chapter is presented from the viewpoint of an administrative user with “Admin” permissions (see [“Account Management”](#) on page 71).

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 470.

Information in this section covers:

- “[Setting Administration Options](#)” on page 104
- “[Entering System Information](#)” on page 105
- “[Configuring Runtime Notifications](#)” on page 105
- “[Configuring Runtime Reporting](#)” on page 106
- “[Managing Accounts](#)” on page 111
- “[Choosing Roles and Protocols](#)” on page 112
- “[Specifying Federation Information](#)” on page 114
- “[Setting System Options](#)” on page 116
- “[Configuring Outbound Provisioning Settings](#)” on page 119
- “[Configuring Auto-Connect Metadata Signing](#)” on page 120
- “[Configuring Auto-Connect Metadata Lifetime](#)” on page 121
- “[Saving and Editing Server Settings](#)” on page 122

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 74).

The screenshot shows the PingFederate administrative console. At the top, there are two tabs: 'Main' and 'Server Settings', with 'Server Settings' being the active tab. Below the tabs is a navigation bar with several menu items: 'System Administration' (marked with a star), 'System Info', 'Runtime Notifications', 'Runtime Reporting', 'Account Management', 'Roles & Protocols', 'Federation Info', and 'System Options'. Under 'System Administration', there are sub-items: 'WS-Trust STS Settings', 'Outbound Provisioning', 'Metadata Signing', 'Metadata Lifetime', and 'Summary'. Below the navigation bar, there is a light blue banner with the text: 'Select the style of application management practiced in your organization.' Underneath this banner, the 'System Administration Style' is shown with two radio buttons: 'Single-user Administration' and 'Multi-user Administration'. The 'Multi-user Administration' radio button is selected.

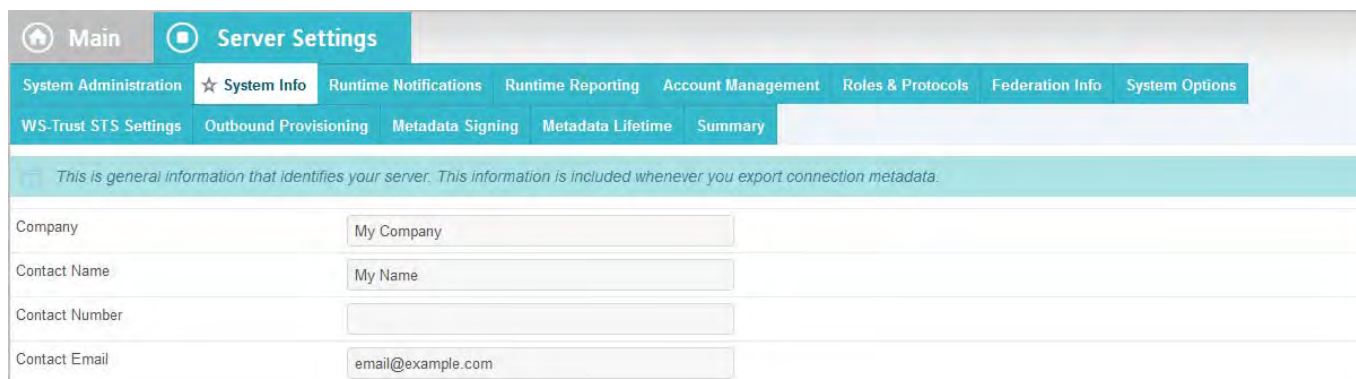
If you choose Single-user Administration, the console is accessible only by using the default administrator ID, for which full privileges are provided. Multi-user Administration (the default) provides role-based access control (see “[Account Management](#)” on page 71).



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

Entering System Information

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



This is general information that identifies your server. This information is included whenever you export connection metadata.

Company	My Company
Contact Name	My Name
Contact Number	
Contact Email	email@example.com

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 50).

Select which server events result in notifications sent via email.

Notification for server licensing events

Email Address

Notification for certificate expiration events

Email Address

Initial Warning Event (days)

Final Warning Event (days)

[Email Server 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 77).

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 107).

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 576).

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 110).



Important: Authentication is required for JMX-client access to PingFederate runtime data (see [“Authentication”](#) on page 233).



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 54). 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 106).

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 56).

To re-enable JMX monitoring:

1. Edit `<pf_install>/pingfederate/bin/run.properties`.
2. Change the value of `provisioner.events.monitor` from `LOG` to `JMX`.
3. Edit `<pf_install>/pingfederate/server/default/conf/jmx.remote.access`.
4. Change the value of `default-jmx-principal` from `readonly` to `readwrite`.
5. Save all changes.
6. Restart PingFederate.



Note: Repeat these steps on each PingFederate server configured to process Outbound Provisioning (see [“Outbound Provisioning for IdPs”](#) on page 35).

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 19 Outbound Provisioning JMX Monitoring Options

Method	Description	Parameter
<pre>viewEvents(Boolean wasSuccessful, String eventTypeStr, String fromDate, String toDate)</pre>	<p>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".</p>	<p><i>wasSuccessful</i> – If true, returns information only on successful transactions; false returns information only on failed transactions; null returns all transactions.</p> <p><i>eventTypeStr</i> – The type of event. Valid values are: CREATE, UPDATE, DISABLE, ENABLE; null or an empty string returns all types.</p> <p><i>fromDate</i> – See Note below.</p> <p><i>toDate</i> – See Note below.</p>
<pre>eventSummaryReport(String fromDate, String toDate)</pre>	<p>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.</p>	<p>See Note below.</p>
<pre>provisioningCycleSummary(String fromDate, String toDate)</pre>	<p>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.</p>	<p>See Note below.</p>
<pre>eventSummaryReportAllData()</pre>	<p>Gets a summary of transaction counts with no time constraints (equivalent to <code>eventSummaryReport</code> with null or empty strings used as parameters).</p>	<p>None.</p>
<pre>eventSummaryRollup()</pre>	<p>Gets a report representing an aggregate of multiple Summary Reports covering the last 0, 1, 2, 7, 30, 60, 90, 180, and 360 days.</p>	<p>None.</p>
<p>Note: Date parameters may be formatted as either <code>yyyy</code>, <code>yyyy-MM-dd</code>, or <code>yyyy-MM-dd HH:mm:ss</code>. A null value or empty string for a date parameter indicates no constraint for that end of the range.</p>		

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 20 Selected Jetty Metrics

MBean	Attributes
<p><code>com.pingidentity.appserver.jetty.server.connector.ssl.nio:runtimesslselectchannelconnector</code></p> <p>(For Jetty connectors including the primary and secondary PingFederate runtime server ports)</p>	<p><code>connections</code> – The total number of TCP connections accepted by the server.</p> <p><code>connectionsDuration*</code> – How long connections are kept open. Maximum, mean, standard deviation, and total accumulated time are available.</p> <p><code>connectionsOpen</code> – The current number of open connections. Maximum is also available (<code>connectionsOpenMax</code>).</p> <p><code>connectionsRequests*</code> – Number of requests processed per connection. Maximum, mean, and standard deviation are available.</p>
<p><code>org.eclipse.jetty.server.handler:statisticshandler</code></p>	<p><code>requests</code> – Total number of requests received.</p> <p><code>requestsActive</code> – Number of requests currently being processed. Max is also available.</p> <p><code>requestTime</code> – Request duration. Maximum, mean, standard deviation, and total accumulated time are available.</p> <p><code>responses1xx</code>, <code>responses2xx</code>, <code>responses3xx</code>, ... – Total number of requests that returned HTTP status codes of 1xx, 2xx, 3xx, etc.</p>
<p><code>org.eclipse.jetty.util.thread:queuedthreadpool</code></p> <p>(Two pools: one for the runtime server, with 200 maximum threads; one for the administrative console, with 20 maximum threads)</p>	<p><code>idleThreads</code> – Number of idle threads currently available.</p> <p><code>threads</code> – Number of threads currently running (including both idle and active).</p> <p><code>minThreads</code> – Minimum number of threads in the pool.</p> <p><code>maxThreads</code> – Maximum number of threads in the pool.</p> <p><code>lowOnThreads</code> – A boolean flag indicating whether the pool is running low on threads.</p>
<p><code>java.lang: Memory</code> <code>java.lang: MemoryPool</code> <code>java.lang: GarbageCollection</code> <code>java.lang: OperatingSystem</code></p>	<p>Various attributes measuring CPU usage and memory.</p>

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 222). (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 104).

USERNAME	USER ADMIN	ADMIN	CRYPTO ADMIN	ACTION	
Administrator	<input type="radio"/> Auditor <input checked="" type="radio"/> Admin	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Deactivate / Change Password



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 74).

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 71). 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 71.

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 469.

Select the role(s) and protocol(s) that you intend to use with your federation partners.

- Enable OAuth 2.0 Authorization Server (AS) role
 - OpenID Connect
- Enable Identity Provider (IdP) role and support the following:
 - SAML 2.0
 - Auto-Connect Profile
 - SAML 1.1
 - SAML 1.0
 - WS-Federation
 - Outbound Provisioning
 - WS-Trust
- Enable Service Provider (SP) role and support the following:
 - SAML 2.0
 - Auto-Connect Profile
 - Attribute Requester Mapping for X.509 Attribute Sharing Profile (XASP)
 - SAML 1.1
 - SAML 1.0
 - WS-Federation
 - WS-Trust
 - Inbound Provisioning
- Enable IdP Discovery role (SAML 2.0 only)

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 15).
- 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 32).
- Also for either role, you can enable the Outbound/Inbound Provisioning option (see “User Provisioning” on page 34).

- ▶ 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 32).



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 120)
 - Metadata Lifetime (see [“Configuring Auto-Connect Metadata Lifetime”](#) on page 121)
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 35.)



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 352). 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 353).

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 36.)

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 373).



Tip: After you make this selection and create XASP IdP connections (see [“Configuring the Attribute Query Profile”](#) on page 322), 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 5, “OAuth Configuration”](#)
- [Chapter 7, “Identity Provider SSO Configuration”](#)
- [Chapter 8, “Service Provider SSO Configuration”](#)
- [Chapter 9, “WS-Trust STS Configuration”](#)
- [“Configuring IdP Discovery”](#) on page 139

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 38).

You can also use a different ID for Auto-Connect transactions (see [“Using Auto-Connect”](#) on page 32).

Main
Server Settings

System Administration
System Info
Runtime Notifications
Runtime Reporting
Account Management
Roles & Protocols
★ Federation Info
System Options

WS-Trust STS Settings
Outbound Provisioning
Metadata Signing
Metadata Lifetime
Summary

You must create a unique identifier for your server for use with your federation partners. A unique identifier is required for each protocol enabled. You will need to communicate this with your partners out-of-band or through metadata exchange. The Base URL is used to construct other URLs in the system and may be used as part of your system ID. The SAML 1.x Source ID defaults to the SHA-1 hash of the SAML 1.x Issuer. Enter an alternate Source ID value if desired.

Base URL	<input type="text" value="https://localhost:9031"/> *
SAML 2.0 Entity ID	<input type="text" value="pingfederate3.default.entityid"/> *
Auto-Connect Entity ID (URL)	<input type="text" value="http://example.com/sso"/>
SAML 1.x Issuer/Audience	<input type="text" value="urn:saml1.example.com.issuer"/> *
SAML 1.x Source ID	<input type="text"/>
WS-Federation Realm	<input type="text" value="urn:wsfederate.example.sso"/> *

Field Descriptions

Field	Description
Base URL	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 63).
SAML 2.0 Entity ID	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: <code>pingidentity.com</code> . The SAML SourceID used for artifact resolution is derived from this ID using SHA1.
Auto-Connect Entity ID	<p>(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 34).</p> <p>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 32).</p>
SAML 1.x Issuer/Audience	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.

Field	Description
SAML 1.x Source ID	(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.
WS-Federation Realm	The URI of the realm associated with the PingFederate server. A realm represents a single unit of security administration or trust.



Note: The fields available on this screen depend on the federation protocols enabled on your server (see “[Choosing Roles and Protocols](#)” on page 112).

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

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 23).

For more information about connection validation, see:

- [“Managing Connection Validation”](#) on page 272 (for SP connections)
- [“Managing IdP Connection Validation”](#) on page 380 (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:

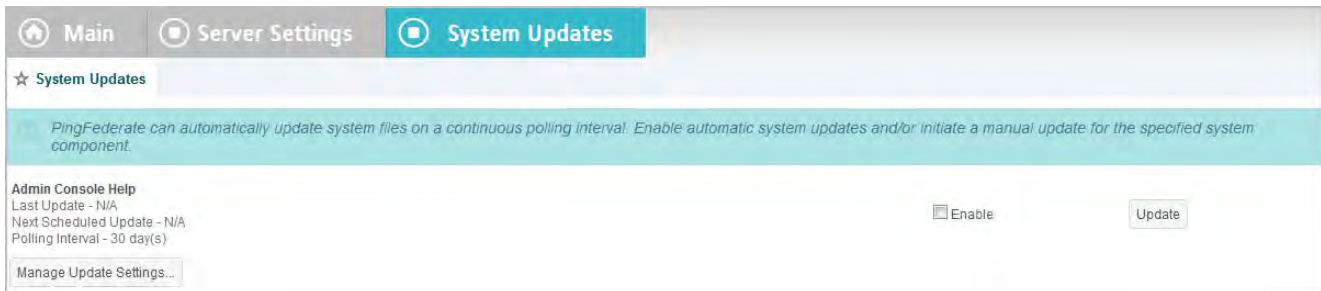
- Leave the default of **Use Last Value** to use the last value in the combined list.
- Select **Use First Value** to use the first value in the combined list.

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.



Each system component displays a series of messages regarding the current status for the update. PingFederate currently supports updating the context-sensitive **Help** system for the administrative console.

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**.

See the next section for more information.

Configuring Update Settings

On this screen you can set the polling interval for automatic updates and indicate proxy server settings (if applicable).

System Update Settings

Specify update settings:

Polling Interval (days) 30 *

Proxy Host MyHost

Proxy Port 6553

Field Descriptions

Field	Description
Polling Interval (days)	The number of days used to determine how often PingFederate checks for new updates. Valid values range from 1 to 90.
Proxy Host	The domain name or IP address of the proxy host.
Proxy Port	The proxy port number. This field is an integer between 1 and 65535.

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 35).

This screen is presented only if Outbound Provisioning is enabled for the IdP federation role (see [“Choosing Roles and Protocols”](#) on page 112).



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.

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 339.)



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`

To configure the internal data store:

1. Select the data store from the drop-down list.
If the data store you want is not shown in the list, then PingFederate is not yet configured to access the store; click **Manage Data Stores** to create a connection to the data store (see [“Managing Data Stores”](#) on page 122).
2. (Optional) Change the Synchronization Frequency value.

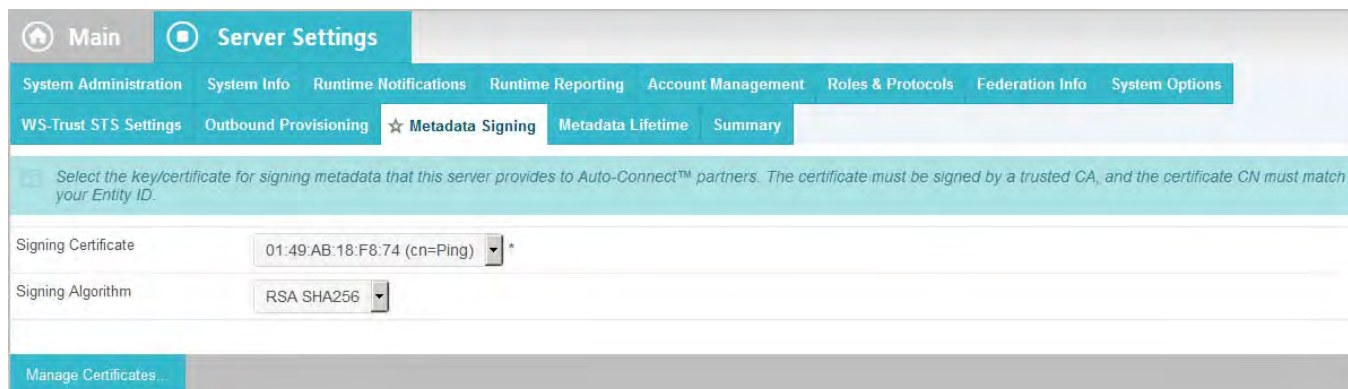
Configuring Auto-Connect Metadata Signing

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 32).

On the Metadata Signing screen, choose a certificate to use for signing the metadata.



Important: The certificate CN must match the domain name associated with the Entity ID (see [“Specifying Federation Information”](#) on page 114).



This screen appears only if Auto-Connect is enabled for either an IdP or SP (see [“Choosing Roles and Protocols”](#) on page 112).

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 227).
2. (Optional) Select the Signing Algorithm from the drop-down list.
The default selection is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection 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 34).

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 112).

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 50). 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 25).

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 292). 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 15). 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 34).

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.

DESCRIPTION	SYSTEM ID	USER	TYPE	LDAP TYPE	ACTION
-------------	-----------	------	------	-----------	--------



Important: You must have current connectivity from PingFederate to a data store in order to create or modify the configuration. If you find that the configuration is not editable, then your connection has been lost due to a system problem not related to the PingFederate server. The problem must be identified and corrected before you can continue.

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 123.
 - For LDAP configuration information see [“Configuring an LDAP Connection”](#) on page 127.
 - For Custom configuration information see [“Configuring a Custom Data Store”](#) on page 133.
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

Field	Description
JDBC URL	<p>(Required) The location of the JDBC database, in the format:</p> <pre>jdbc:mysql://databaseservername/ databasename</pre> <pre>jdbc:sqlserver:// databaseservername;databaseName=databasename</pre> <pre>jdbc:oracle:thin:@databaseservername:databasename</pre> <p>where <i>databaseservername</i> is the DNS host name (or IP) of the server hosting the database, and <i>databasename</i> is the name of a database on that server.</p> <p>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:</p> <pre>?autoReconnect=true</pre> <p>For more information, see the field description for Validate Connection SQL below.</p>
Driver Class	<p>(Required) The name of the driver class used to communicate with the source database. For example, <code>com.mysql.jdbc.Driver</code>, <code>com.microsoft.sqlserver.jdbc.SQLServerDriver</code>, or <code>oracle.jdbc.OracleDriver</code>. This class should be supplied by the database software vendor in a JAR file, which must be present in the <code>pingfederate/server/default/lib</code> directory.</p>
Username	<p>(Required) The name that identifies the user when connecting to the database.</p>

Field	Description
Password	The password needed to access the database.
Validate Connection SQL	<p>(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.</p> <p>If a SQL statement is not provided here, PingFederate may not be able to reconnect to the database if the connection is broken.</p> <p>Note: To use this feature for MySQL, you must also add a query parameter to the JDBC URL (see information in field description above).</p> <p>Important: Ensure that the SQL statement is valid for your database. Examples:</p> <p>(For Oracle or MySQL) <code>SELECT 1 from dual</code></p> <p>(For SQL Server) <code>SELECT getdate()</code></p>
Mask Values in Log (Checkbox)	Determines whether all attribute values returned from this data store will be masked in PingFederate log files (see “Attribute Masking” on page 25).
Allow Multi-Value Attributes (Checkbox)	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.

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.

- (Optional) Select Mask Values in Log.
For information, see [“Attribute Masking”](#) on page 25.
- (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 126).
- Click **Next**.



Note: PingFederate tries to connect to the database at this point. If it cannot, there may be a problem with your settings.

- On the Summary screen, click **Done**.
- 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.

Field Descriptions

Field	Description
Minimum Pool Size	The smallest number of database connections in the connection pool for the given data store.
Maximum Pool Size	The largest number of database connections in the connection pool for the given data store.
Blocking Timeout (ms)	The amount of time a request waits to get a connection from the connection pool before it fails.
Idle Timeout (min)	The length of time the connection can be idle in the pool before it is closed.

To reach this screen for editing:

- Click **Data Stores** on the Main Menu.
- Click the Data Store Description link on the Manage Data Stores screen.
- 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.

Field Descriptions

Field	Description
Hostname(s)	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. Note: If more than one Hostname is entered, each server must be accessible using the same User DN and Password (or via Bind Anonymously).

Field	Description
LDAP Type	<p>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 35).</p> <p>Identifying the LDAP type permits PingFederate to configure many provisioning settings automatically (see “Modifying Source Settings” on page 344).</p> <p>The LDAP type is also used to enable password-change messaging between AD and PingFederate (see “HTML Form Adapter Configuration” on page 549).</p> <p>Tip: If you are using Outbound Provisioning and your LDAP server is <i>not</i> AD or Oracle, you may wish to define a custom LDAP Type to streamline the provisioning configuration (see “Defining an LDAP Type” on page 129).</p>
Bind Anonymously (Checkbox)	(Optional) Username and password are not required (see procedure below).
User DN	<p>The username credential required to access the data store.</p> <p>Important: The user must have permission to search the directory for user-account information. For security, this user should have read-only access. When connecting to an AD LDAP server running on Microsoft Windows Server 2008 R2 (or higher), enter a User account; do not use a Computer account.</p>
Password	The password credential required to access the data store.
Use LDAPS (Checkbox)	<p>(Optional) Connects to the LDAP data store using LDAPS. This selection applies equally to any secondary servers specified in Hostname(s).</p> <p>Important: We recommend that all LDAP connections be secured using LDAPS.</p>
Mask Values in Log (Checkbox)	(Optional) Determines whether all attribute values returned from this data store will be masked in PingFederate log files (see “Attribute Masking” on page 25).

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:



Note: The fields referenced in the following steps are detailed in the Field Descriptions table above.

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.



Note: If you choose an anonymous binding, ensure that this access level provides permission to search the directory for user-account information.

4. (Optional) Select Use LDAPS.



Important: We recommend that all LDAP connections be secured using LDAPS.

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 130).
7. Click **Next**.



Note: PingFederate attempts to connect to the data store 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.

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 35).

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 344).

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
```

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 344).

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.

The screenshot displays the 'Advanced LDAP Options' configuration page. At the top, there are navigation tabs: 'Main', 'Manage Data Stores', and 'Data Store'. Below these, the 'Advanced LDAP Options' section is active, with a sub-tab for 'LDAP Binary Attributes'. A teal banner at the top of the configuration area contains the text: 'Manage LDAP connection-pooling settings on this screen as needed.' The configuration items are as follows:

- Test Connection on Borrow
- Test Connection on Return
- Create New Connections if Necessary
- Verify LDAPS Hostname
- Minimum Connections: 10 *
- Maximum Connections: 100 *
- Maximum Wait (Milli): -1 *
- Time Between Eviction (Milli): 60000 *
- Read Timeout (Milli): 3000 *
- Connection Timeout (Milli): 3000 *

An 'Apply Defaults...' button is located at the bottom left of the configuration area.

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.

Field	Description
Test Connection on Borrow (Checkbox)	Indicates whether objects are validated before being borrowed from the pool.
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.
Verify LDAPS Hostname	Indicates whether to verify the hostname of the LDAP server matches the Subject (CN) or one of the Subject Alternative Names from the certificate. Important: We recommend to verify LDAPS hostname in all LDAP connections.
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 <code>maxThreads</code> 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 <code>maxThreads</code> 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).

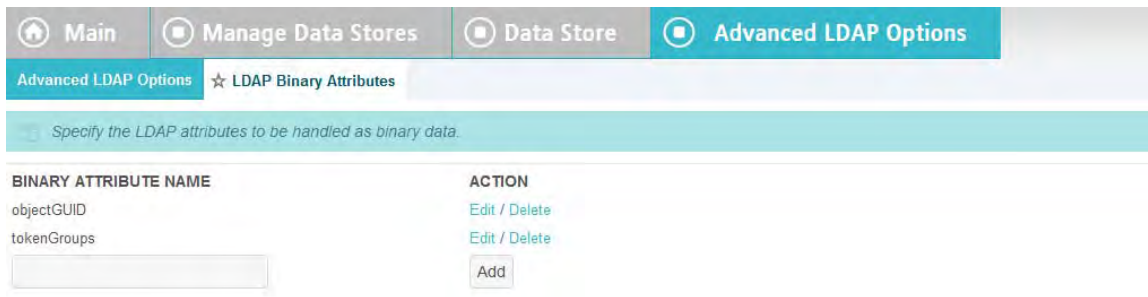
Field	Description
Time Between Eviction (Milli)	The frequency, in milliseconds, that the evictor cleans up the connections in the pool. A value of -1 disables the evictor.
Read Timeout (Milli)	The maximum number of milliseconds a connection waits for a response to be returned before producing an error. 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.
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.

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 25.
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.

FIELD NAME	FIELD VALUE	DESCRIPTION
PRIMARY SOAP ENDPOINT	https://www.example.com/soape	Primary Endpoint
SECONDARY SOAP ENDPOINT	https://www.example.com/soape	Secondary Endpoint
USERNAME	admin	Username to allow access to the datastore.
PASSWORD	*****	Password associated to username.

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 19). 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>/pingfederate/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 123).
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>/pingfederate/server/default/data/config-store`, open the file:

```
org.sourceid.saml20.service.impl.  
    AccountLinkingServiceDBImpl.xml
```

Between the XML tags for the item named `PingFederatedDSJNDIName`, 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 127).
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>/pingfederate/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>/pingfederate/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:

 - `PingFederatedDSJNDIName`: 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](http://msdn.microsoft.com/en-us/library/ms676900%28v=VS.85%29.aspx) 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 134), PingFederate uses its internal Hypersonic database by default to maintain persistent grants (if any) for the OAuth AS—Persistent grants can result from some OAuth use cases (see “[Persistent vs. Transient Grants](#)” on page 12 for more information).



Important: When persistent grants are expected, administrators should consider changing this configuration to use an external secured database for production standalone deployments. For server clustering, an external database is required, since the Hypersonic database cannot be shared across other PingFederate engine nodes.

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 tables for grant storage:

- ▶ Run the table-setup scripts for your database server provided in the directory:
`<pf_install>/pingfederate/server/default/conf/access-grant/sql-scripts`

If scripts are not provided for your database server, 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 123).
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>/pingfederate/server/default/data/config-store`, open the file:
`org.sourceid.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.
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

By default, PingFederate stores OAuth clients entered into the system through the administrative console (see [“Client Management”](#) on page 190) or the REST-based application programming interface (see [“PingFederate Administrative API”](#) on page 610) in XML files. Alternatively, it can be configured to use a centralized database for OAuth client records, allowing the records to be managed via the administrative console, the REST-based application programming interface, or an OAuth Client Web Service (see [“OAuth Client Management Service”](#) on page 598).



Tip: 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. Administrators should consider changing this configuration to use an external secured database for production standalone deployments.



Important: For server clustering, an external database is required, since the Hypersonic database cannot be shared across other PingFederate engine nodes.

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>/pingfederate/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>/pingfederate/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 123).
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>/pingfederate/server/default/data/config-store`, open the file:
`org.sourceid.oauth20.domain.ClientManagerJdbcImpl.xml`
Change the value of the `item` named `PingFederateDSJNDIName` 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 142). 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 112). Then click **IdP Discovery** under System Settings on the Main Menu to reach this screen.



For an overview of this SAML 2.0 profile, see [“IdP Discovery”](#) in the “Supported Standards” chapter of *Getting Started*.

► 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 112).

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 112).
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.)

The screenshot shows the 'Configure IdP Discovery' screen with the 'Common Domain Service' tab selected. The form contains the following fields:

- Base URL of the PingFederate common domain service: *
- Pass Phrase:
- Confirm Pass:

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 112).
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 112).

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.

The screenshot shows the 'Configure IdP Discovery' screen. The navigation tabs are 'Main', 'IdP Discovery', and 'Configure IdP Discovery'. Under 'Configure IdP Discovery', there are sub-tabs: 'Domain Cookie Settings', 'Common Domain Service', 'Local Common Domain Server', and 'Summary'. The 'Local Common Domain Server' tab is active. A message at the top says: 'Please provide configuration information for this server to act as the common domain service.' Below this are four input fields: 'Common Domain' with the value '.example.com', 'Cookie Lifetime (days)' with the value '30', 'Pass phrase' with masked characters, and 'Confirm Pass' with masked characters.

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 112).
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 139).

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 566).

► *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 275).

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.sourceid.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 415).

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:

<code>IdpIdCookieName</code>	The name of the IPRC set by the SP installation (default: <code>IdPId</code>). Note that the cookie name cannot contain any of the following characters: <code>&</code> , <code>></code> , <code><</code> , comma, semicolon, space.
<code>IdpIdCookieLifetimeInDays</code>	The lifetime for the cookie (default: 365 days, maximum: 24,855 days). The browser will delete the cookie when the period is expired.
<code>ShowIdpSelectionList</code>	If set to <code>true</code> (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 <code>false</code> , the SP installation generates an error page.

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.

Configuring Redirect Validation

Several SP adapters can be configured to pass security tokens or other user credentials from the PingFederate SP server 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 or SLO links to start an SSO or SLO 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 and SLO transactions to ensure that the designated target resource exists in a domain controlled by the SP through a list of configurable URLs.



Note: Validating target resource for SSO is only applicable for IdP connections, adapter-to-adapter mappings and SAML 2.0 IdP Discovery.

The following default target URLs are always allowed:

- The default target URL for any IdP connections (see “[Configuring Default Target URLs \(Optional\)](#)” on page 423)
- The default target URL for any adapter-to-adapter mappings (see “[Configuring a Default Target URL \(Optional\)](#)” on page 152)
- The SP default URL for successful SSO (see “[Configuring Default URLs](#)” on page 372)
- The IdP default URL for successful SLO (see “[Configuring a Default URL and Error Message](#)” on page 264)

They do not need to be entered into the list manually.



Tip: PingFederate is also capable of validating the error resource (`InErrorResource`) parameter for the same reason. For more information, see “[IdP Endpoints](#)” on page 563, “[SP Endpoints](#)” on page 567 and “[System-Services Endpoints](#)” on page 576.



Important: PingFederate enables target resource validation for SSO and SLO, as well as error resource URLs, and requires HTTPS in new installations by default.

For backward compatibility, PingFederate Upgrade Utility does not enable these options if they were not selected in the previous PingFederate installation. Although optional, it is strongly recommended to enable target and error resource validation (including the HTTPS option) and to enter all expected resources to prevent unauthorized access.

Require HTTPS	Valid Domain Name (leading wildcard *. allowed)	Valid Path (leave blank to allow any path)	Allow Any Query/Fragment	TargetResource for SSO	TargetResource for SLO	InErrorResource
<input checked="" type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

To reach this screen:

- ▶ Click **Redirect Validation** on the Main Menu.

To enable target resource validation for SSO and/or SLO:

- ▶ Select the SSO and/or SLO checkbox(es).

To enable error resource validation:

- ▶ Select the Enable InErrorResource validation checkbox.

To enter expected resources:

1. 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.

2. Enter the domain or IP address containing the expected target resource under Valid Domain Name.

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 the resource (case-sensitive) under Valid Path. 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 resource.



Note: When selected, no query parameter and/or fragment is allowed in the path.

Select the TargetResource for SSO, TargetResource for SLO and/or InError Resource checkbox(es) to indicate the resource type(s).

Click **Add**.

3. Repeat the previous step to add additional entries as needed.
4. Click **Save**.

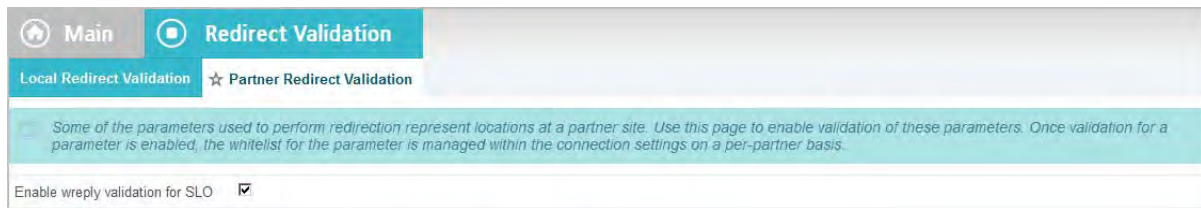
Managing Partner Redirect Validation

Some of the parameters used to perform redirection represent locations at a partner site—for example, the wreply parameter in WS-Federation. To protect against session token hijacking via open redirections, PingFederate provides an option to validate wreply for SLO. Once enabled, the whitelist for the parameter is managed within the connection settings on a per-partner basis. PingFederate amalgamates the entries from all active WS-Federation connections and validates wreply against the consolidated list.



Important: PingFederate enables wreply validation for SLO in new installations by default.

For backward compatibility, PingFederate Upgrade Utility does not enable this option if it was not selected in the previous PingFederate installation. Although optional, it is strongly recommended to enable wreply validation for SLO and to specify the allowed domains and paths for each WS-Federation IdP connection to prevent unauthorized access (see [“Specifying a Service URL \(WS-Federation\)”](#) on page 417).



To reach this screen:

1. Click Redirect Validation on the Main Menu.
2. Click Partner Redirect Validation.

To enable wreply validation for SLO:

1. Select the Enable wreply validation for SLO checkbox.
2. Click **Save**.

To disable wreply validation for SLO:

1. Clear the Enable wreply validation for SLO checkbox.
2. Click **Save**.

IdP-to-SP Bridging

The IdP-to-SP Bridging links on the Main Menu (under Server Configuration) provide access to advanced federation settings.

This chapter covers:

- [“Adapter-to-Adapter Mapping”](#) on page 147
- [“Token Exchange Mapping”](#) on page 155
- [“Connection Mapping Contracts”](#) on page 162



Note: The information in this chapter is presented from the viewpoint of an administrative user with “Admin” permissions (see [“Account Management”](#) on page 71).

Adapter-to-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 35).

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:

- “Configuring IdP Adapters” on page 250
- “Configuring SP Adapters” on page 362

Managing Mappings

On the Manage Mappings screen you can add, modify, or delete Adapter-to-Adapter Mappings.

ADAPTER-TO-ADAPTER MAPPINGS	ACTION
Source Instance: - SELECT -	Target Instance: - SELECT -
<input type="button" value="Add Mapping..."/>	

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**.

The screenshot shows the 'Mapping Configuration' screen with the 'License Assignment' tab selected. A message reads: 'Select a license group to assign the mapping to.' Below this, a 'License Group' dropdown menu is set to 'A (01/01/2015-12/31/2017, 2)'.

Configuring Attribute Lookup for Adapter-to-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 adapter-to-adapter configuration allows you to configure one or more data stores to look up attributes and to set up search parameters.

The screenshot shows the 'Mapping Configuration' screen with the 'Attribute Sources & User Lookup' tab selected. A message reads: 'Optionally, you may select a data store from which to look up any attributes required by the SP Adapter Contract but not available directly from the IdP Adapter. Click Next to continue if no data store is needed.' Below this is a table with columns 'DESCRIPTION', 'TYPE', and 'ACTION'. At the bottom left, there is a button labeled 'Add Attribute Source...'.

- ▶ 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 151).

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 122).



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:

Data Store Type	Manual Section
JDBC	<ul style="list-style-type: none"> • “Selecting a JDBC Database Table and Columns” on page 295 • “Configuring a Database Filter (WHERE Clause)” on page 297
LDAP	<ul style="list-style-type: none"> • “Configuring an LDAP Directory Search” on page 299 • “Configuring an LDAP Filter” on page 302
Custom	<ul style="list-style-type: none"> • “Configuring Custom Source Filters” on page 304 • “Selecting Custom Source Fields” on page 304

Identifying Target Application (Optional)

Use this screen to enter the name of the target application and the URL of the application icon, accessible through the IdP Adapter interface (`IdpAuthenticationAdapterV2`) in the PingFederate Java SDK. (For more information about the SDK, see [SDK Developer's Guide](#).) Both fields are optional.

To complete the configuration:

- ▶ Enter an Application Name and an Application Icon URL.

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).

SP ADAPTER CONTRACT	SOURCE	VALUE	ACTIONS
E-mail Address	- SELECT -		None available
First Name	- SELECT -		None available
Last Name	- SELECT -		None available
subject	- SELECT -		None available

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 Adapter-to-Adapter Mapping”](#) on page 149).

- 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 613). 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 149) 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 372).



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 576).

Optional, you can specify a default target URL for this adapter-to-adapter mapping configuration. Entering a URL in the Default Target URL field overrides any SP Default URL SSO setting.

Default Target URL

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 25). This token authorization can be used to restrict who can SSO to protected resources.

PingFederate can evaluate various criteria to determine whether users are authorized to access SP resources. Use this optional screen to configure the criteria for use with this conditional authorization.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
-SELECT-	-SELECT-	-SELECT-			Add

Show Advanced Criteria

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.

- 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 Adapter-to-Adapter Mapping”](#) on page 149).

- Mapped Attributes – Select to access the required attributes.
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 93).



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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.

- 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 618. 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 Adapter-to-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.

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 469). 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 475
- [“Configuring Token Generators”](#) on page 509

Managing Token Mappings

On the Manage Mappings screen you can add, modify, or delete token-to-token mappings.

TOKEN-TO-TOKEN MAPPINGS	ACTION
'UsernameTokenProcessor' to 'Saml2TokenGenerator'	Delete

Source Instance: - SELECT - Target Instance: - SELECT - Add Mapping...

To reach this screen:

- ▶ Click **STS Token Exchange Mapping** under Advanced Federation 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. When multiple IdP token processors and/or SP token generators are configured, you must provide the `TokenProcessorId` and/or the `TokenGeneratorId` query parameter(s) in the request (see ["System-Services Endpoints"](#) on page 576).

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.

- 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 122).



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:

Data Store Type	Manual Section
JDBC	<ul style="list-style-type: none"> • “Selecting a JDBC Database Table and Columns” on page 295 • “Configuring a Database Filter (WHERE Clause)” on page 297
LDAP	<ul style="list-style-type: none"> • “Configuring an LDAP Directory Search” on page 299 • “Configuring an LDAP Filter” on page 302
Custom	<ul style="list-style-type: none"> • “Configuring Custom Source Filters” on page 304 • “Selecting Custom Source Fields” on page 304

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).

TOKEN GENERATOR CONTRACT	SOURCE	VALUE	ACTIONS
SAML_SUBJECT	Token	username	None available

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 157).

- 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 613). 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:

```
${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 157) 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

page 25). This token authorization can be used to restrict who can access protected Web services.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.



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. 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.

Choose Expression and then click **Edit** to enter an expression (see “Using the OGNL Edit Screen” on page 618). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 613.

For syntax and examples, see sections under “Constructing Expressions” on page 614.)

- 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 “Choosing a Data Store for Token Mapping” on page 157).

- 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.



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 take into account minor differences that might be present in the DN syntax (for example, whether spaces are present after commas).

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.

For more information on SOAP, see the World Wide Web Consortium's [Simple Object Access Protocol](http://www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507) (www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507).



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 613).

- Use the in-line editor box to enter the OGNL expression.

For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.

- 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 618.

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.

Connection Mapping Contracts

PingFederate uses two connections to bridge an identity provider to a service provider:

- An IdP connection where end users authenticate and PingFederate (the federation hub) is the SP
- An SP connection to the target application where PingFederate (the federation hub) is the IdP

It fuses these two connections together by a connection mapping contract—a connection mapping contract is the medium that carries user attributes from the identity provider to the service provider.

Each connection mapping contract comes with one default attribute (*subject*). You can extend the contract to include additional attributes as needed. In most federation hub use cases, you configure PingFederate to pull attribute values from inbound assertions into the connection mapping contract in an IdP connection and to push those values from the connection mapping contract into the outbound assertions through an SP connection. For advanced use cases, you have the option to configure the IdP and/or SP connections to look up values from multiple data store instances.

When bridging one identity provider to one service provider, you need to create one connection mapping contract and associate the contract with both the IdP connection and the SP connection.

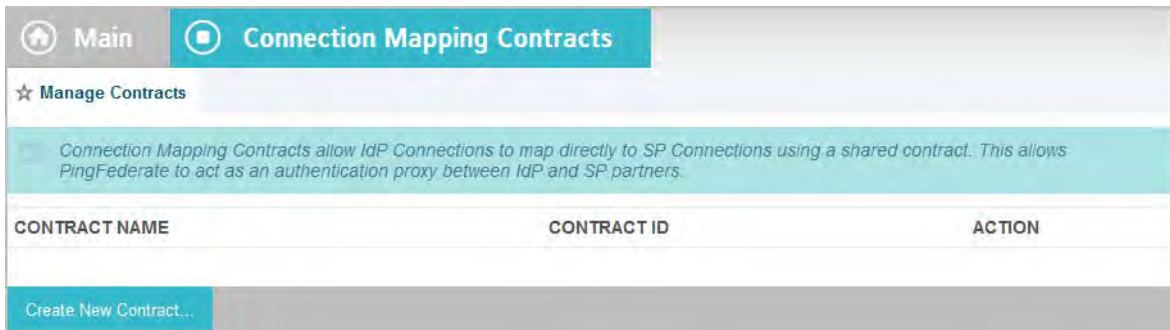
When bridging one identity provider to multiple service providers, you need to create a connection mapping contract per service provider because each service provider likely requires a different set of attributes. Map all the connection mapping contracts into the IdP connection. Add the respective connection mapping contract to each SP connection to the service provider.

When bridging multiple identity providers to one service provider, you likely need only one contract unless the service provider requires a different set of attributes from each identity provider. Add the connection mapping contract to the SP connection as well as the applicable IdP connections.

(See “[Federation Hub](#)” on page 42 for more information.)

Managing Contracts

On the Manage Contracts screen, you can add, modify, or delete connection mapping contracts.



To reach this screen:

- ▶ Click **Connection Mapping Contracts** under IdP-to-SP Bridging on the Main Menu.

To add a contract:

- ▶ Click **Create New Contract**.

To edit a contract:

1. Click the contract name.

To delete a contract:

1. Click **Delete** under Actions for the mapping that you want to delete.



Note: This option does not appear if the connection mapping contract is associated with at least one IdP or SP connection. To enable deletion, click **Check Usage** to locate the connection(s) and change the mappings as needed.

2. Click **Save**.

Editing Contract Info

On the Contract Info screen, you can enter or modify the contract name .

To create a new contract:

1. Enter a new contract name.
2. Click **Next**.

To modify an existing contract:

1. Edit the contract name as needed.
2. Click **Next**.

Defining Contract Attributes

On the Contract Attributes screen, you can enter or modify the set of user attributes to map between an SP connection and an IdP connection.

To extend the contract:

1. Enter the attribute name in the text box.
Attribute names are case-sensitive and must correspond to the attribute names expected by your SP and IdP connections.
2. Click **Add**.
3. Click **Next**.

To modify an existing contract:

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.

3. Click **Next**.

To delete an attribute:

1. Click **Delete** under Action for the attribute.

2. Click **Next**.

Using the Connection Mapping Contracts Summary Screen

When you have finished configuring connection mapping contracts, 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 Contracts 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 411.

Enabling the OAuth AS

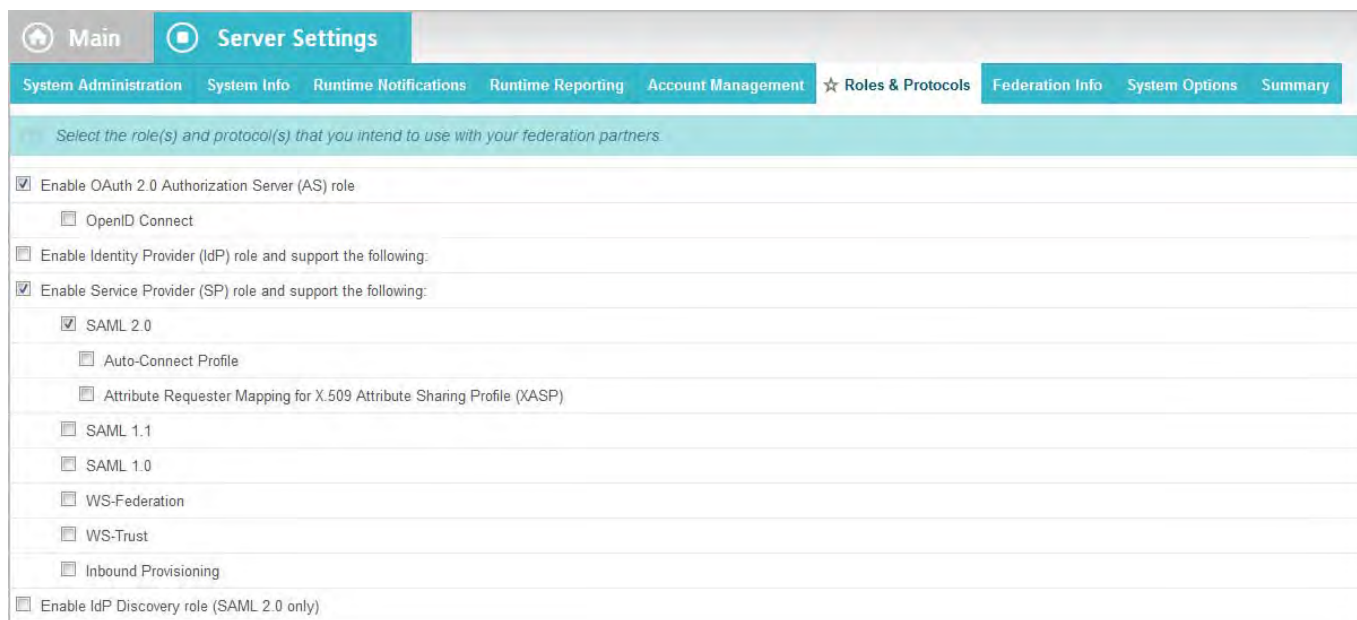
You can enable the OAuth AS when you first install PingFederate (see [“Running PingFederate for the First Time”](#) in the “Installation” chapter of *Getting Started*). If you have already installed PingFederate or are upgrading to a new version, use the following procedure.



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.



3. Select Enable OAuth 2.0 Authorization Server (AS) role.
4. (Optional) Click OpenID Connect to enable that feature (see “[OpenID Connect](#)” on page 15).
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 13). (Additional optional mapping scenarios are provided for SPs as part of the IdP Connection task flow—see “[Configuring OAuth Attribute Mapping](#)” on page 411 and “[Configuring an OAuth SAML Grant IdP Connection](#)” on page 209.)



At a minimum from the Main Menu, an administrator must:

1. Configure Authorization Server Settings (see “[Authorization Server Settings](#)” on page 169).
2. Configure settings for OAuth clients (see “[Client Management](#)” on page 190).
3. Configure how [access tokens](#) are handled, including setting up an Attribute Contract (see “[Access Token Management](#)” on page 175).

4. Configure a default attribute mapping from persistent grants to access tokens (see [“Access Token Mapping”](#) on page 204).



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 13), 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 15).



Note: This option appears only after the OpenID Connect protocol is enabled in Server Settings (see [“Choosing Roles and Protocols”](#) on page 111).

- ▶ 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 204).
 - [“Resource-Owner Credentials Mapping”](#) on page 195
Use this option to map persistent grants based on authentication handled by PingFederate via Password Credential Validators (see [“Validating Password Credentials”](#) on page 235).
 - [“IdP Adapter Mapping for OAuth”](#) on page 200
Use this option to map persistent grants based on user authentication information provided by PingFederate IdP adapters (see [“Configuring IdP Adapters”](#) on page 250).
 - [“Configuring OAuth Attribute Mapping”](#) on page 411
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 563). 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 168.



Tip: Information about the Asynchronous Front-Channel Logout endpoint is also available in the OpenID Connect metadata (see “[OpenID Connect Metadata Endpoint](#)” on page 590). 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 169).
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 191).

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 607.

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 191).
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 184).
3. In the Authorization Server Settings screen, modify the **Session Revocation Lifetime** value as needed (see [“Authorization Server Settings”](#) on page 169).



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 550).

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.

Main
Authorization Server Settings

★ **Manage Settings**

Provide general configuration and policy for the PingFederate Authorization Server.

Scope Settings

Default Scope Description

Scope Value	Scope Description	Action
<input style="width: 90%;" type="text"/>	<input style="width: 90%;" type="text"/>	<input type="button" value="Add"/>

Scope Group Value	Scope Group Description	Sub Scopes	Action
<input style="width: 90%;" type="text"/>	<input style="width: 90%;" type="text"/>		<input type="button" value="Add"/>

Authorization Code Policy Settings

Authorization Code Timeout (seconds) *

Authorization Code Entropy (bytes) *

Refresh Token and Persistent Grant Settings

Persistent Grant Lifetime (blank for indefinite) Days ▼

Refresh Token Length (characters) *

Roll Refresh Token Values (default policy)

Minimum Interval to Roll Refresh Tokens (hours) *

Reuse Existing Persistent Access Grants for Grant Types Implicit

Authorization Code
 Resource Owner Password Credentials

Bypass Authorization for Previously Approved Persistent Grants

Allow Unidentified Clients to Make Resource Owner Password Credentials Grants

Allow Unidentified Clients to Request Extension Grants

Persistent Grant Extended Attributes

Attribute	Action
<input style="width: 90%;" type="text"/>	<input type="button" value="Add"/>

OAuth Administrative Web Services Settings

Password Credential Validator - SELECT - ▼

OpenID Connect Settings

Track User Sessions for Logout

Revoke User Session on Logout

Session Revocation Lifetime (minutes) *

Field Descriptions

Field	Description
Default Scope Description	<p>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.</p> <p>Tip: 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 98).</p>
Scope Value	<p>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.</p> <p>Tip: For OpenID Connect, you can change the message presented to users for their authorization by entering <code>openid</code> as a Scope Value and the message you want under Scope Description.</p> <p>Important: Also for OpenID Connect, to use the following key attributes in authorization policies, you <i>must</i> add them in this row, along with Scope Descriptions (see “Configuring OpenID Connect Policies” on page 184).</p> <ul style="list-style-type: none"> • <code>profile</code> • <code>email</code> • <code>address</code> • <code>phone</code> <p>Note: By default, scopes are available to all OAuth clients unless access is restricted on a per-client basis (see “Configuring a Client” on page 191).</p>
Scope Description	<p>A description of the Scope Value. This description appears when the user is prompted for authorization.</p> <p>Tip: 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 98).</p>
Scope Group Value	<p>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</p>

Field	Description
Scope Group Description	<p>A description of the scope group value. This description appears when the user is prompted for authorization.</p> <p>Tip: 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 98).</p>
Authorization Code Timeout (seconds)	The amount of time (in seconds) that an authorization code is considered valid.
Authorization Code Entropy (bytes)	The length (in bytes) of the authorization code returned to the OAuth Client.
Persistent Grant Lifetime (blank for indefinite)	<p>(Integer) Indicates the number of days or hours the OAuth AS stores persistent grants. Leave blank to maintain grants indefinitely.</p> <p>Note: You can also set persistent grant expiration for specific OAuth clients, overriding this global configuration (see “Client Management” on page 190).</p>
Refresh Token Length (characters)	The number of characters the OAuth AS uses to generate the refresh token returned to the client.
Roll Refresh Token Values (default policy) (checkbox)	<p>When selected, the OAuth AS generates a new refresh token value when a new access token is obtained.</p> <p>Note: New refresh token values are not issued during the defined interval (see the Minimum Interval to Roll Refresh Tokens box below).</p> <p>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).</p>
Minimum Interval to Roll Refresh Tokens (hours)	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.

Field	Description
Reuse Existing Persistent Access Grants for Grant Types (checkboxes)	<p>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.</p> <p>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.</p> <p>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 <i>if</i> the checkbox Bypass Authorization for Previously Approved Persistent Grants is also selected.</p>
Bypass Authorization for Previously Approved Persistent Grants (checkbox)	<p>When selected, consent from the user is requested only once (see “OAuth User-Facing Pages” on page 98). 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 <i>and</i> when the Reuse Existing Persistent Access Grants for Grant Types checkbox is selected.</p> <p>Tip: On a per-client basis, user consent may be bypassed completely, overriding this setting (see “Configuring a Client” on page 191).</p>
Allow Unidentified Clients to Request Resource Owner Password Credentials Grants (checkbox)	<p>When selected, allows Resource Owners to obtain access tokens without defining a client.</p>
Allow Unidentified Clients to Request Extension Grants (checkbox)	<p>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).</p>
Persistent Grant Extended Attributes	<p>Extend persistent grants to include additional attributes from your authentication systems.</p> <p>Tip: This is useful for Default access-token attribute mappings, because you can fulfill the access-token contracts by the extended attributes. For more information, see “Access Token Mapping” on page 204 and “Token Attribute Contract Fulfillment” on page 206.</p>

Field	Description
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 598). The service cannot be used if a validator is not provided.
Track User Sessions for Logout	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 167).
Revoke User Session on Logout	When selected, PingFederate revokes the corresponding sessions on logout (see “Back-Channel Session Revocation” on page 168). Note: PingFederate always adds the session to the revocation list, even if a logout error occurs.
Session Revocation Lifetime (minutes)	The amount of time (in minutes) until the revoked sessions are removed from the revocation list for optimal performance. Important: The value should match or exceed the maximum session lifetime of the relying parties (see “Back-Channel Session Revocation” on page 168).

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.

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 190).

2. Click **Save** on the Authorization Server Settings screen.

Access Token Management

PingFederate supports multiple access token management instances. This capability allows you (the administrator) to configure different access token policies and attribute contracts for different OAuth clients. It also provides a means to control validation of access tokens to a certain RS.

Clients can specify an access token management instance by providing the access token manager ID (`access_token_manager_id`) or a resource URI (`aud`) in their requests to the PingFederate AS.

When defining an access token management instance, you can customize various settings, including the token format, lifetime and attribute contract for this instance. You can also limit the access token management instance to a list of resource URIs and/or clients in an access control list (ACL). For example, you can use the ACL to limit which clients can obtain access tokens from a particular access token management instance.

You can also add the desired RS clients to the ACL, such that these are RS clients can ensure that the access token they have received was issued by a specific access token management instance. They do this by specifying which instance will be used in validating the token (via the `access_token_manager_id` or `aud` parameter) they want to use in the token validation request.

At runtime, the PingFederate AS uses the following rules to determine which access token management instances it should use:

1. Limit the eligible access token management instances to those that are available in the context of the request. For most requests, these are instances that have an attribute mapping defined in Access Token Mapping. For OAuth SAML Grant requests, it is the set of instances for which a mapping is defined in the IdP Connection. Furthermore, the client ACL (if configured) can also limit which access token management instances are eligible.
2. If the request comes with an `access_token_manager_id` or `aud`, the PingFederate AS uses the information to determine the applicable access token management instance.
3. If the request does not come with either parameter, for OpenID Connect clients (where the scope value contains `openid` in their requests), the PingFederate AS uses the access token manager specified by the OpenID Connect policy of the client.

4. If the request comes with neither of the two parameters nor the `openid` scope, the PingFederate AS uses the default access token manager of the client (if configured) or the default instance defined for the installation.

If no match can be found in the eligible list, the PingFederate AS aborts the request. For more information about the access token manager ID (`access_token_manager_id`) and resource URI (`aud`) parameters, see the “Access Token Management parameters” sections under “Token Endpoint” on page 579 and “Authorization Endpoint” on page 586.

Managing Access Token Management Instances

Use this screen flow to specify how the PingFederate AS manages OAuth access tokens.

INSTANCE NAME	INSTANCE ID	TYPE	PARENT NAME	ACTION
ATM Staging	AtmStaging	JSON Web Tokens		Delete / Set as Default
ATM Worldwide	AtmWw	Internally Managed Reference Tokens		Delete (Check Usage) / Default
ATM North America	AtmNa	Internally Managed Reference Tokens	ATM Worldwide	Delete / Set as Default

To configure a new instance:

- ▶ Click **Create New Instance**.

To edit an existing access token management instance:

- ▶ Click the Instance Name and click the step you need to change.

To delete an access token management 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.

Defining an Access Token Management Instance

Use this screen to create a new or to edit an existing access token management instance.

To create a new instance:

1. Enter an Instance Name and Instance Id. The Instance Id may not contain spaces or underscores.
2. Choose the plug-in type of the access token management instance.

Type 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](https://pingidentity.com/support) (pingidentity.com/support)

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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

4. Click **Next**.

To edit an existing access token management instance:

1. Edit the Instance Name as needed.
2. (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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

3. Click **Next**.

Configuring a Token-Management Instance

This configuration varies depending on the Type of the access token manager. The following sections provide information for token plug-ins bundled with PingFederate:

- [“Configuring Reference-Token Management”](#) (next)
- [“Configuring JSON-Token Management”](#) on page 179

Configuring Reference-Token Management

On this screen, you can make changes to preset default settings for Reference Tokens (see [“Token Models and Management”](#) on page 10) as well as set an optional maximum token lifetime.

Main
Access Token Management
Create Access Token Management Instance

Type
★ Instance Configuration
Access Token Attribute Contract
Resource URIs
Access Control
Summary

Complete the configuration necessary to issue and validate access tokens. This configuration was designed into, and is specific to, the selected Access Token Management plugin.

FIELD NAME	FIELD VALUE	DESCRIPTION
TOKEN LENGTH	<input type="text" value="28"/> *	Defines how many alphanumeric characters make up an access token.
TOKEN LIFETIME	<input type="text" value="120"/> *	Defines how long, in minutes, an access token is valid.
LIFETIME EXTENSION POLICY	<input type="text" value="No Extension"/>	Dictates which tokens are eligible for lifetime extension. Similar to a session inactivity timeout, the lifetime period of an access token can be reset each time the token is validated at the AS (subject to the values defined for the Lifetime Extension Threshold Percentage and the Maximum Token Lifetime).
MAXIMUM TOKEN LIFETIME	<input type="text"/>	(Optional) Defines an absolute maximum token lifetime, in minutes, for use with the Lifetime Extension Policy. An access token's lifetime cannot be extended beyond this setting.
LIFETIME EXTENSION THRESHOLD PERCENTAGE	<input type="text" value="30"/> *	Defines the percentage of a token's lifetime remaining before the lifetime is actually extended, which can improve cluster performance.

Show Advanced Fields

► Make changes as needed (see **Field Descriptions** below), or click **Next** to continue.



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

Field Descriptions

Field	Description
Token Length	The number of characters the AS uses to define the token reference. Increasing the length will enhance token security, if desired. (Maximum: 256)
Token Lifetime	The amount of time (in minutes) that an access token is considered valid.
Lifetime Extension Policy	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.
Maximum Token Lifetime	(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.

Field	Description
Lifetime Extension Threshold Percentage	<p>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.</p> <p>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.</p>
Advanced Fields	
Mode for Synchronous RPC	<p>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.</p> <p>Synchronous RPC calls occur when a node receives a verification request for a token it does not recognize and for token issuance.</p>
RPC Timeout	Timeout between cluster nodes during synchronous communication. Recommended setting is from 100 milliseconds to 1000 (1 second).

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 and Management”](#) on page 10).

Main
Access Token Management
Create Access Token Management Instance

Type
★ Instance Configuration
Access Token Attribute Contract
Resource URIs
Access Control
Summary

Complete the configuration necessary to issue and validate access tokens. This configuration was designed into, and is specific to, the selected Access Token Management plugin.

A JSON Web Token (JWT) Bearer Access Token Management Plug-in that enables PingFederate to issue (and optionally validate) cryptographically secure self-contained OAuth access tokens.

SYMMETRIC KEYS (A group of keys for use with HMAC-based integrity algorithms)

KEY ID (An identifier for the given key)	KEY (Hex-encoded Key)	Action
Add a new row to 'Symmetric Keys'		

CERTIFICATES (A group of certificates and their corresponding public/private key pairs for use with signatures)

KEY ID (An identifier for the given key)	CERTIFICATE (Requires RSA key length of at least 2048 bits)	Action
Add a new row to 'Certificates'		

FIELD NAME	FIELD VALUE	DESCRIPTION
TOKEN LIFETIME	120 *	Defines how long, in minutes, an access token is valid.
JWS ALGORITHM	-- Select One --	+ The HMAC or signing algorithm used to protect the integrity of the token. For HMAC, the active symmetric key must be selected below. For RSA, the active signing certificate must be selected.
ACTIVE SYMMETRIC KEY ID	-- Select One --	The Key ID of the key to use when producing JWTs using an HMAC-based algorithm.
INCLUDE KEY ID HEADER PARAMETER	<input checked="" type="checkbox"/>	Indicates whether the Key ID (kid) header parameter will be included in the token, which can help identify the appropriate key during verification.
ACTIVE SIGNING CERTIFICATE KEY ID	-- Select One --	The Key ID of the key pair and certificate to use when producing JWTs using an RSA-based algorithm.
INCLUDE X.509 THUMBPRINT HEADER PARAMETER	<input type="checkbox"/>	Indicates whether the X.509 Certificate Thumbprint (x5t) header parameter will be included in the token, which can help identify the appropriate public key during verification of asymmetrically signed tokens.
CLIENT ID CLAIM NAME	client_id	The name of the JWT claim used to represent the OAuth Client ID (omitted, if blank).
SCOPE CLAIM NAME	scope	The name of the JWT claim used to represent the scope of the grant (omitted, if blank).

Manage Signing Certificates ...
Show Advanced Fields

The configuration provides for token security using either symmetric keys or asymmetric signing-certificate keys (see “[Digital Signing and Decryption Keys and Certificates](#)” on page 227). Multiple entries are allowed for either signing mechanism to facilitate rollover of keys when they expire.



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

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).
3. Click **Next**.

Field Descriptions

Field	Description
Token Lifetime (Required)	The amount of time, in minutes, that an access token is considered valid.
JWS Algorithm (Required)	The hash-based message authentication code (HMAC) or RSA signing algorithm used to protect the integrity of the token.
Active Symmetric Key ID	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.)
Include Key ID Header Parameter	When selected (the default), the Key ID is used in the <code>kid</code> header parameter for the token.
Active Signing Certificate Key ID	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.)
Include X.509 Thumbprint Header Parameter	When selected, the X.509 Certificate Thumbprint is used in the <code>x5t</code> header parameter for the token.
Client ID Claim Name	The name of a JWT claim used to represent the OAuth Client ID.
Scope Claim Name	The name of a JWT claim used to represent the scope of the grant.
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 (<code>iss</code>) claim in the JWT.
Audience Claim Value	The value of the Audience (<code>aud</code>) claim in the JWT.
JWT ID Claim Length	Indicates the number of characters of the JWT ID (<code>jti</code>) 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 <code>https://<pf_host>:<port>/ext/oauth/x509/kid?v=<id></code>

Field	Description
Publish Thumbprint X.509 URL	Indicates whether certificates will be made accessible by thumbprint at <code>https://<pf_host>:<port>/ext/oauth/x509/x5t?v=<base64url encoded SHA-1 thumbprint></code>

Defining the Access Token Attribute Contract

On this screen, create a list of attributes to be referenced in an OAuth access token. You must enter at least one attribute.

Main | Access Token Management | Create Access Token Management Instance

Type | Instance Configuration | **Access Token Attribute Contract** | Resource URIs | Access Control | Summary

Provide the names of the attributes that will be carried in (or referenced by) the OAuth access token.

EXTEND THE CONTRACT ACTION

Add



Note: If this is a child instance, select the override checkbox to modify the configuration.

To add an attribute:

1. Enter the attribute name in the text box.
2. Click **Add**.

Add additional attributes as needed. Attribute names are case-sensitive and must correspond to the attribute names expected by the Resource Server.

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.

Click **Next** when finished editing attributes.

Configuring Resource URIs

Specify a list of resource URIs which can be used to select this Access Token Management instance.

Main | Access Token Management | Create Access Token Management Instance

Type | Instance Configuration | Access Token Attribute Contract | **Resource URIs** | Access Control | Summary

Specify a list of base resource URIs which can be used to select this Access Token Management instance.

RESOURCE URIS	ACTION
<input type="text"/>	Add



Note: If this is a child instance, select the override checkbox to modify the configuration.

To add a resource URI:

1. Enter the value in the text box.
2. Click **Add**.

Enter additional resource URIs as needed. The resource URIs must correspond to the resource expected by the Resource Server.

To modify a resource URI:

1. Click **Edit** under Action for the resource URI.
2. Make the change and click **Update**.

To delete a resource URI:

- Click **Delete** under Action for the resource URI.

Click **Next** when finished editing resource URIs. (To undo the deletion, click **Undelete**.)

Defining Access Control

Use this screen to enable access control by OAuth clients.



Note: If this is a child instance, select the override checkbox to modify the configuration.

To enable access control:

1. Select the Restrict Allowed Clients checkbox.
2. Select a client under Allowed Clients and click **Add**.
3. Select additional clients as needed.

To remove a previously allowed client:

- Click **Delete** next to the applicable client. (To undo the deletion, click **Undelete**.)

To disable access control:

- Clear the Restrict Allowed Clients checkbox.

Click **Next** when finished configuring access control.

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 15). 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 168).



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 111).



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. Select an Access Token Manager.
4. (Optional) Change the default ID Token Lifetime.
Values can be between 0 and 65,535
5. (Optional) Select the Include Session Identifier in ID Token checkbox to add a session identifier in the ID Tokens.
6. (Optional) Select the Include User Info in ID Token checkbox to include additional attributes in the ID Tokens.



Tip: Alternatively, OAuth clients can obtain additional attributes from the UserInfo endpoint (see “[OpenID Connect Metadata Endpoint](#)” on page 590).

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.

The screenshot shows the 'Policy Management' interface with the 'Policy' tab selected. The 'Attribute Contract' sub-tab is active. A teal informational banner at the top states: 'The required Attribute Contract here consists of a user identifier ("sub"). You may extend the contract to include attributes that will be returned to OAuth clients in response to requests received at the PingFederate UserInfo endpoint. The preset extended-contract list contains OpenID Connect standard attributes: add, edit, or delete items in this list as needed for this policy.'

Below the banner, the 'ATTRIBUTE CONTRACT' section shows a text input field containing 'sub'. The 'EXTEND THE CONTRACT ACTION' section is a table with the following rows:

ATTRIBUTE CONTRACT	ACTION
sub	
address.locality	Edit / Delete
address.postal_code	Edit / Delete
address.region	Edit / Delete
address.street_address	Edit / Delete
email	Edit / Delete
name	Edit / Delete
phone_number	Edit / Delete
profile	Edit / Delete

At the bottom of the table, there is an empty text input field and an 'Add' button.

To add an attribute:

- ▶ Enter the attribute name in the text box and click **Add**.

To modify an attribute name:

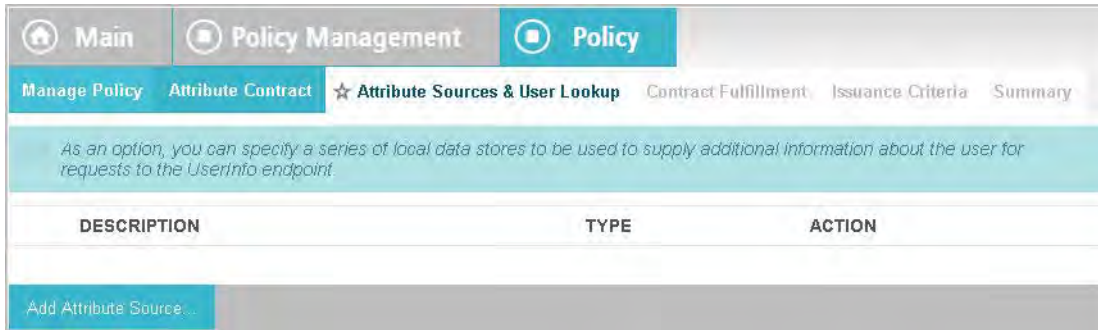
1. Click **Edit** under Action for the attribute.
2. Make the change and click **Update**.

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, enter an Attribute Source Id and an Attribute Source Description. Then, select an Active Data Store and click **Next** to 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 216. 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.

ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS
address.locality	- SELECT -		None available
address.postal_code	- SELECT -		None available
address.street_address	- SELECT -		None available
email	- SELECT -		None available
family_name	- SELECT -		None available
given_name	- SELECT -		None available
name	- SELECT -		None available
nickname	- SELECT -		None available
phone_number	- SELECT -		None available

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: 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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 218, [“Defining a JDBC Location for OAuth”](#) on page 217, or [“Configuring OAuth Custom Source Filters”](#) on page 220).

- 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 613). 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 175).

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 25).

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.



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 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.



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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618.

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 137).



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 598).



Tip: For detailed developer information for client applications, see [“OAuth 2.0 Endpoints”](#) on page 579.

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.

Main **Client Management** **Client**

★ **Manage Client**

Manage the configuration and policy information about a client.

Client ID

Client Authentication

None

Client Secret

Client Certificate

Client TLS Certificate Issuer

Client TLS Certificate Subject DN

Extract Subject DN From Certificate

Name

Description

Redirect URIs

REDIRECTION URIS	ACTION
<input type="text"/>	<input type="button" value="Add"/>

Logo URL

Bypass Authorization Approval

Restrict Scopes

Allowed Grant Types

Authorization Code

Resource Owner Password Credentials

Refresh Token

Implicit

Client Credentials

Access Token Validation (Client is a Resource Server)

Extension Grants

Default Access Token Manager

Persistent Grants Expiration

Use Global Setting

Grants Do Not Expire

*

Refresh Token Rolling Policy

Use Global Setting Don't Roll Roll

OpenID Connect

ID Token Signing Algorithm

Policy

Grant Access to Session Revocation API

PingAccess Logout Capable

LOGOUT URIS

LOGOUT URIS	ACTION
<input type="text"/>	<input type="button" value="Add"/>

Field Descriptions

Field	Description
Client ID	(Required) A unique identifier the client provides to the Resource Server to identify itself. This identifier is included with every request the client makes.
Client Authentication	<p>A selection other than None is required only for the Client Credentials grant type (see “Grant Types” on page 11).</p> <p>When required, select <i>either</i>:</p> <ul style="list-style-type: none"> • Client Secret for Basic authentication. <ul style="list-style-type: none"> 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. <p><i>Or:</i></p> <ul style="list-style-type: none"> • Client Certificate for mutual SSL/TLS authentication—recommended for application configurations where security policies prohibit storing passwords. <ul style="list-style-type: none"> Note: Mutual SSL/TLS authentication requires the use of a secondary PingFederate SSL port (see “Changing Configuration Parameters” on page 80). Enter information into the following required fields: <ul style="list-style-type: none"> Client TLS Certificate Issuer – Select a Trusted CA from the list (see “Trusted Certificate Authorities” on page 222). 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	<p>(Required) A descriptive name for the client instance. This name appears when the user is prompted for authorization.</p> <p>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 98).</p>
Description	<p>A description of what the client application does. This description appears when the user is prompted for authorization.</p> <p>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 98).</p>

Field	Description
Redirection URIs	<p>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).</p> <p>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.</p> <p>For example: <code>https://*.company.com/*</code></p> <p>Important: 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 <code>redirect_uri</code> parameter (see "OAuth 2.0 Endpoints" on page 579).</p>
Logo URL	<p>The location of the logo used on user-facing OAuth grant authorization and revocation pages (see "Customizing User-Facing Screens" on page 93). (For best results with the installed HTML templates, the recommended size is 72 x 72 pixels.)</p>
Bypass Authorization Approval	<p>When selected, resource-owner approval for client access is assumed—the authorization consent page is not presented to the user for this client.</p> <p>Use this setting, for example, when you want to deploy a trusted application and authenticate end users via an IdP adapter or IdP connection.</p>
Restrict Scopes	<p>Restricts this client's access to specific scopes or scope groups.</p> <p>When selected, a list appears of all scopes defined within the PingFederate AS. Select the scopes or scope groups available for this client.</p> <p>Note: 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 169). If a client requests a specific scope (not selected in the list), an error is returned.</p>
Allowed Grant Types	<p>Available grant types that this client is allowed to use.</p>
Default Access Token Manager (optional)	<p>Select a default Access Token Manager for this client (see "Access Token Management" on page 175).</p>
Persistent Grants Expiration	<p>Allows an administrator to override the Persistent Grant Lifetime set globally for the OAuth AS (see "Authorization Server Settings" on page 169).</p>

Field	Description
Refresh Token Rolling Policy	<p>Select Don't Roll or Roll to override the Roll Refresh Token Values setting on the Authorization Server Settings screen (see “Authorization Server Settings” on page 169).</p> <p>Leave Use Global Setting selected to default to the Roll Refresh Token Values setting on the Authorization Server Setting screen.</p>
OpenID Connect	<p>Note: These options are displayed only when OpenID Connect is enabled in Server Settings (see “Choosing Roles and Protocols” on page 111).</p> <p>ID Token Signing Algorithm (Optional): Select the signing algorithm for the ID Token. The default algorithm is <code>RSA using SHA-256</code>.</p> <p>Policy (Optional): Choose a specific OpenID Connect policy (see “Configuring OpenID Connect Policies” on page 184).</p> <p>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 607).</p> <p>Note: If the Track User Sessions for Logout checkbox is selected in the Authorization Server Settings screen, there are two additional settings:</p> <p>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 167).</p> <p>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 167).</p> <p>For more information about the logout endpoint in PingAccess, see OpenID Connect Endpoints in the user guide for PingAccess.</p>

Resource-Owner Credentials Mapping

This configuration allows you to map attributes based on validated user credentials into the `USER_KEY` and extended attributes for a persistent grant (see [“Authorization Server Settings”](#) on page 169 and [“Mapping OAuth Attributes”](#) on

page 13). 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 235).

To create a mapping:

- ▶ 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 235).

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 Attribute Sources and User Lookup

Attribute sources are specific data store or directory locations containing information that may be needed to fulfill the USER_KEY and/or the extended attribute values of persistent grants. This configuration is optional.

- ▶ If you are not using a data store, click **Next** and see the next section, “Resource-Owner Contract Fulfillment”.

To configure an attribute source:

1. Click **Add Attribute Source**.
2. On the Data Store screen, enter an Attribute Source Id and an Attribute Source Description. Then, select an Active Data Store and click **Next** to 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 216. When complete, click **Next** on this screen and see the next section.

Resource-Owner Contract Fulfillment

On this screen, you map values into the `USER_KEY` and extended attributes for a persistent grant. Use this mapping for the Resource Credential grant type (see “Grant Types” on page 11).



Note: The `USER_KEY` values must be unique across all end users because it is the identifier to store and to retrieve persistent grants. For example, if you have two Active Directory domains, the `sAMAccountName` value of an end user in one domain may collide with that of another end user in the other domain. In this scenario, you can map Subject DN to the `USER_KEY`.

Main				Resource Owner Credentials Mappings				Resource Owner Credentials Mapping			
Data Store				★ Contract Fulfillment				Issuance Criteria Summary			
Select a Source and Value to map into each item in the Contract list.											
CONTRACT		SOURCE		VALUE		ACTIONS					
USER_KEY		- SELECT -				None available					

Map each attribute 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 618). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 613. For syntax and examples, see sections under “Constructing Expressions” on page 614.)

- 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 218, “Defining a JDBC Location for OAuth” on page 217, or “Configuring OAuth Custom Source Filters” on page 220).

- 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 613). 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.
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 25). This token authorization can be used to restrict who can access an OAuth-protected resource.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.



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 or an extended attribute.
 - 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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618.

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

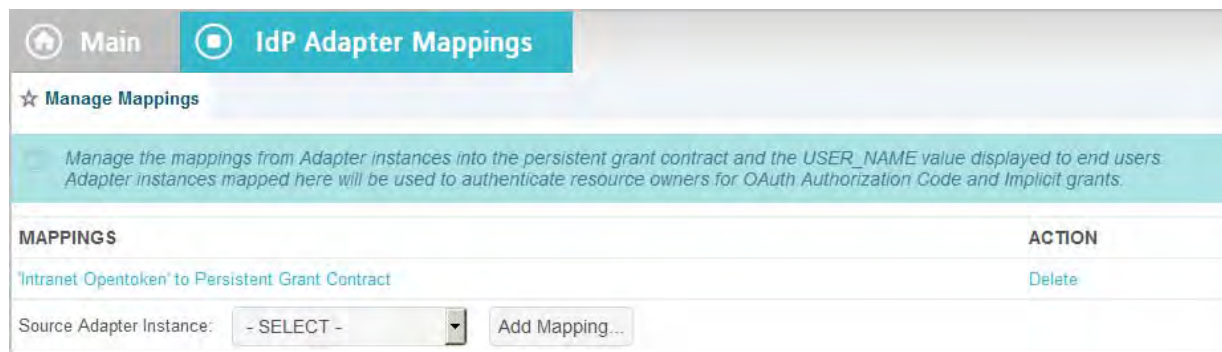
When you have finished the configuration, you can review it on the Summary screen.

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` and extended attributes for a persistent grant, as well as the `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 250.)

Use this mapping for Authorization Code and Implicit grant types (see “[Grant Types](#)” on page 11).



To create a mapping:

- ▶ Select a Source Instance from the drop-down list and click **Add Mapping**.

To edit an existing mapping:

- ▶ Click its link and then on the Summary screen click the heading over the information that needs updating.

IdP Adapter Attribute Sources and User Lookup

Attribute sources are specific data store or directory locations containing information that may be needed to fulfill the `USER_KEY` and/or extended attribute values of persistent grants, as well as the `USER_NAME` presented to the end user for authorization permission. This configuration is optional.

- ▶ If you are not using a data store, click **Next** and see the next section, “[Grant Contract Fulfillment](#)”.

To configure an attribute source:

1. Click **Add Attribute Source**.
2. On the Data Store screen, enter an Attribute Source Id and an Attribute Source Description. Then, select an Active Data Store and click **Next** to 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 216. When complete, click **Next** on this screen and see the next section.

Grant Contract Fulfillment

On this screen, you map values into the `USER_KEY` and extended attributes for a persistent grant, as well as the `USER_NAME` for the user's display name on the authorization page.



Note: The `USER_KEY` values must be unique across all end users because it is the identifier to store and to retrieve persistent grants. For example, if you have two Active Directory domains, the `sAMAccountName` value of an end user in one domain may collide with that of another end user in the other domain. In this scenario, you can map Subject DN to the `USER_KEY`.

Main IdP Adapter Mappings IdP Adapter Mapping			
Attribute Sources & User Lookup		★ Contract Fulfillment	Issuance Criteria Summary
Select a Source and Value to map into each item in the Contract list.			
CONTRACT	SOURCE	VALUE	ACTIONS
USER_KEY	- SELECT -		None available
USER_NAME	- SELECT -		None available

Map each attribute 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.

- 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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 218, [“Defining a JDBC Location for OAuth”](#) on page 217, or [“Configuring OAuth Custom Source Filters”](#) on page 220).

- 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 613). 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.
Choose (or enter) a Value.
2. 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 25). This token authorization can be used to restrict who can access an OAuth-protected resource.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.

- 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 or an extended attribute.
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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618.

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 182).

When mapping a Default context, you define how the OAuth AS maps values into the attributes based on the persistent-grant USER_KEY and any extended attributes (see [“Authorization Server Settings”](#) on page 169).

When a specific context is selected, you can also map attributes from the selected context, namely the chosen IdP adapter, credential validator, or IdP connection (configured with OAuth Attribute Mapping), into the access tokens.

The mapping used at runtime depends on the authentication context of the original grant. If the authentication context results a match, the OAuth AS uses that specific mapping; otherwise, it uses the default mapping for the applicable access token manager.

CONTEXT	TOKEN MANAGER	ACTION
Default	JSON Web Tokens	Delete
IdP Adapter: HtmlFormMarketing	ATM Worldwide	Delete
IdP Connection: ACME IdP Company	JSON Web Tokens	Delete
Validator: PCV for North America	ATM North America	Delete

Context: Access Token Manager:



Note: This option appears on the Main Menu only after Access Token Management is configured (see [“Access Token Management”](#) on page 175).

To create a token mapping:

1. Select a Context from the drop-down list.

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 235)
- User key from the OAuth IdP adapter-mapping configurations (see [“IdP Adapter Mapping for OAuth”](#) on page 200)
- IdP-connection OAuth mappings (see [“Configuring OAuth Attribute Mapping”](#) on page 411)

2. Select an Access Token Manager.

3. Click **Add Mapping**.

To edit an existing mapping:

- Click its link and then on the Summary screen click the heading over the information that needs updating.

Access Token Attribute Sources and User Lookup

Attribute sources are specific data store or directory locations containing information that may be needed to fulfill the token attribute contract. This configuration is optional.

If you are not using a data store, click Next and see the next section, “[Token Attribute Contract Fulfillment](#)”.

To configure an attribute source:

1. Click **Add Attribute Source**.
2. On the Data Store screen, enter an Attribute Source Id and an Attribute Source Description. Then, select an Active Data Store and click **Next** to 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 216. When complete, click **Next** on this screen and see the next section.

Token Attribute Contract Fulfillment

On this screen, you map values into the token attribute contract (see “[Defining the Access Token Attribute Contract](#)” on page 182). These are the attributes that will be included or referenced in the access token.

CONTRACT	SOURCE	VALUE	ACTIONS
uid	- SELECT -		None available

Map each attribute to fulfill the Token Attribute Contract from one of these Sources:

- Adapter/Password Credential Validator/IdP Connection
If you selected a specific IdP adapter, password credential validator or IdP connection under Context in the previous screen, you have the option to map attributes from that specific authentication system. Select the corresponding Context in this screen and choose the desired attribute under Value.
- 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 618). (If the Expression selection is not listed, then the feature is not enabled—see “[Enabling and Disabling Expressions](#)” on page 613. For syntax and examples, see sections under “[Constructing Expressions](#)” on page 614.)

- Persistent Grant
When you make this selection, the associated Value drop-down list is populated by the USER_KEY and extended attributes 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 218, “[Defining a JDBC Location for OAuth](#)” on page 217, or “[Configuring OAuth Custom Source Filters](#)” on page 220).
- 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 613). 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.
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 25). 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

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.



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 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` and extended attributes 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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.

- 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 618.

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.

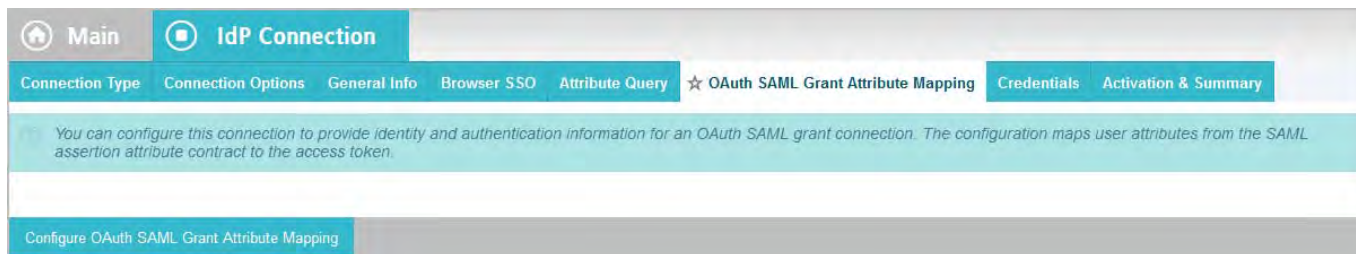
To enable OAuth SAML Grant for a new connection, or to add the capability to an existing connection:

- ▶ On the Connection Type screen, select OAuth SAML Grant (see [“Choosing an IdP Connection Type”](#) on page 381).



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 111) and configure an access token plug-in (see [“Access Token Management”](#) on page 175).

When the option is enabled, the configuration starts on the OAuth SAML Grant Attribute Mapping screen.



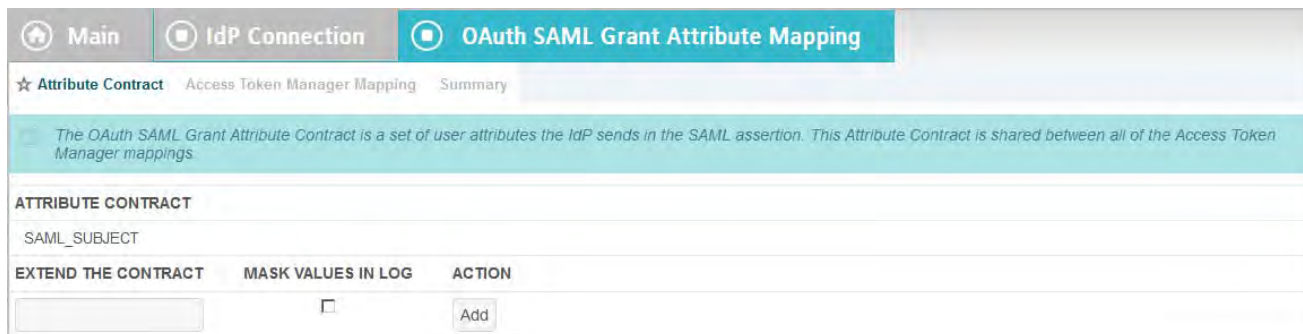
► To continue, click **Configure OAuth SAML Grant Attribute Mapping**.

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 identify 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 25).



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 Access Token Manager Mappings

Use this screen to associate one or more Access Token Managers with this connection to define how access tokens are created.

To create a new access token manager mapping:

- ▶ Click **Create New Access Token Manager Mapping**.

To edit an existing access token manager mapping:

- ▶ Click on the Access Token Manager and click the step you need to change.

To delete an access token manager mapping:

1. Click **Delete** next to the Access Token Manager. (To undo the deletion, click **Undelete**.)
2. Click **Save** to confirm the deletion.

Selecting an Access Token Manager Instance

Use this screen to select an Access Token Manager instance to associate with this connection. Attributes are mapped from the connection's contract into the Access Token Manager's contract to define the resulting content of created access token.

To select a Access Token Manager Instance:

- ▶ Select an Access Token Manager from the drop-down menu.

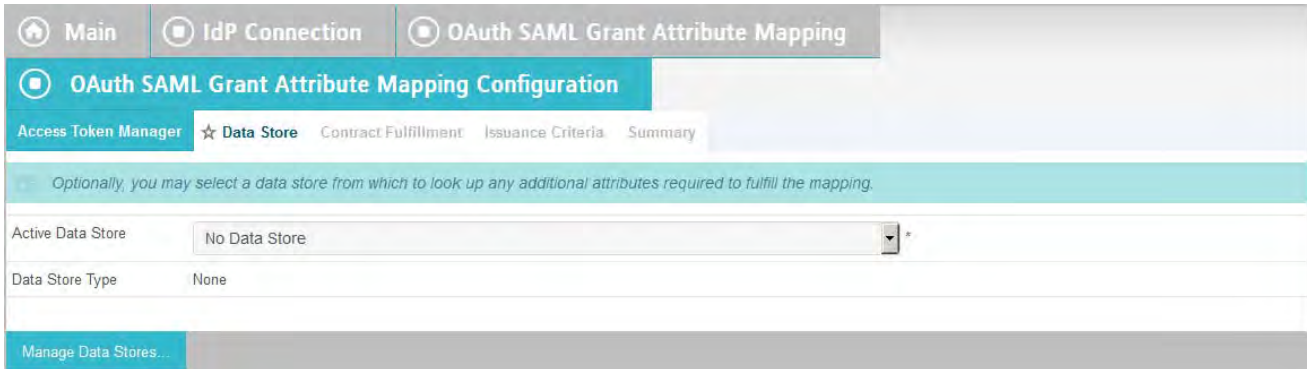


Tip: If the Access Token Manager you need is not available, click **Manage Access Token Management Instances** to define one or more instances you need for this connection.

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 216.

- ▶ If you do not need additional attributes from a data store, just click **Next** on the Data Store screen.

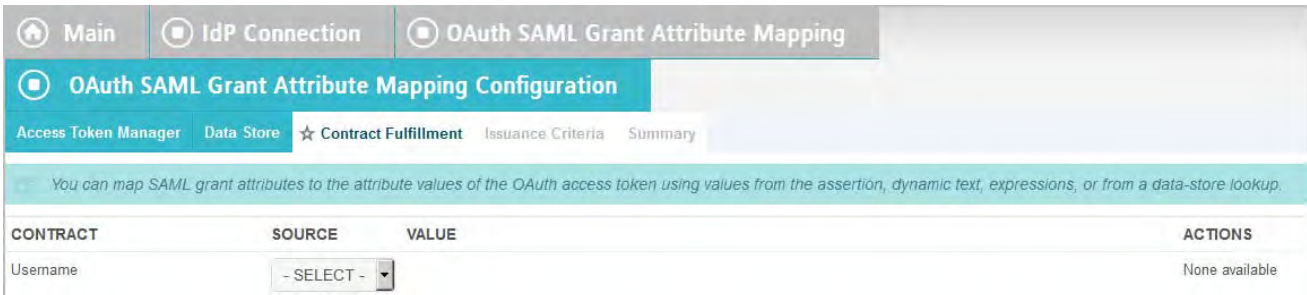


To reach this screen for editing:

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 204).



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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 217 or [“Configuring an OAuth Database Filter \(WHERE Clause\)”](#) on page 218).

- 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 613). 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 25). This token authorization can be used to restrict who can access an OAuth-protected resource.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.
 - Mapped Attributes – Select to access the output of the mapping into the access token.
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.

- (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.

- Click **Add**.
- 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.

- (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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618.

- 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 Attribute Mapping 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 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](#)

A similar configuration is also used for attribute mapping in an IdP connection (see “[Configuring OAuth Attribute Mapping](#)” on page 411) and when configuring an OAuth SAML Grant connection (see “[Configuring an OAuth SAML Grant IdP Connection](#)” on page 209).

- ▶ 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 122).

2. See the following sections, depending on the type of data store:

Data Store Type	Related Section
JDBC	<ul style="list-style-type: none">• “Defining a JDBC Location for OAuth” on page 217• “Configuring an OAuth Database Filter (WHERE Clause)” on page 218
LDAP	<ul style="list-style-type: none">• “Searching LDAP for OAuth Mapping” on page 218• “Configuring an LDAP Filter for OAuth Mapping” on page 219
Custom	<ul style="list-style-type: none">• “Configuring OAuth Custom Source Filters” on page 220• “Selecting OAuth Custom Source Fields” on page 220

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

and

dev.mysql.com/doc/refman/5.1/en/option-files.html

Field Descriptions

Field	Description
Schema	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.
Table	The name of the table contained in the database. Use the drop-down to change the table.
Columns to return from SELECT	Displays selected table columns. Select the columns associated with the desired attributes you would like to return from the JDBC query.

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).



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 the Access Token Attribute Contract”](#) on page 182). Then determine if sufficient information is available from the mapping context.

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 217).

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.
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

Field	Description
Base DN	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.
Search Scope	Determines the node depth of the query. Select Subtree, One level or Object.
Root Object Class	The class containing the attributes you want.
Attributes to return from search	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.

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.



Note: When connecting to Microsoft Active Directory, if you choose the memberOf attribute, an optional checkbox, Nested Groups, appears on the right. Select this checkbox if you want PingFederate to query for groups the end users belong to directly as well as indirectly through nested group membership (if any) under the Base DN.

For example, suppose you have three groups under the Base DN, namely Canada, Washington and Seattle. Seattle is a member of Washington. Ana Smith is an end user and a member of Seattle. If the Nested Groups checkbox is selected, when PingFederate queries for Ana’s memberOf attribute values, the expected results are Seattle and Washington. (When the Nested Groups checkbox is not selected (the default), the expected result is Seattle.)

For Oracle Directory Server, choose isMemberOf under Attribute for nested group membership. For more information, see [documentation about isMemberOf from Oracle](http://docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm) (docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm).

5. Repeat the last step for other attributes as needed.



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 218).

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.

Security Management

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 233



Note: This information is presented from the viewpoint of an administrative user with “Crypto Admin” permissions (see [“Account Management”](#) on page 71).

Certificate Management

PingFederate administrators manage certificates via the Certificate Management section under Server Configuration on the Main Menu.



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.

SERIAL	SUBJECT DN	EXPIRES	STATUS	ACTION
44:DC:CD:D7	CN=localhost, OU=Brian Campbell, O=PingIdentity, L=Denver, ST=CO, C=US	Tue Dec 27 11:35:03 MST 2033	Valid	Export Delete

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

administrative console and for incoming SSL/TLS connections at runtime (see “Setting Administration Options” on page 104).

The screenshot shows the 'Certificate Management' section of an administrative console. It features a navigation bar with 'Main' and 'Certificate Management' tabs. Below the navigation is a sub-header 'Manage SSL Server Certificates' and a descriptive text: 'Establish and maintain the SSL certificate your server presents to incoming SSL connections.' A table lists certificate details:

SERIAL	SUBJECT DN	EXPIRES	KEY DETAILS	STATUS	ACTIVE	ACTION
01.49:AA:9E:70:84	CN=localhost, OU=Development, O=PingIdentity, L=Denver, ST=CO, C=US	Wed Nov 13 11:25:50 PST 2024	RSA 2048	Valid	<input checked="" type="checkbox"/>	Activate for Runtime Server - Certificate already active Activate for Admin Console - Certificate already active Export Certificate Signing Delete (Check Usage)

At the bottom, there are buttons for 'Create New...' and 'Import...'.

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.
2. Select **Certificate Only** on the Export Certificate screen.
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 Server Certificates screen.

To delete a certificate:

1. Click **Delete** under Action for the certificate you want to delete.



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**.

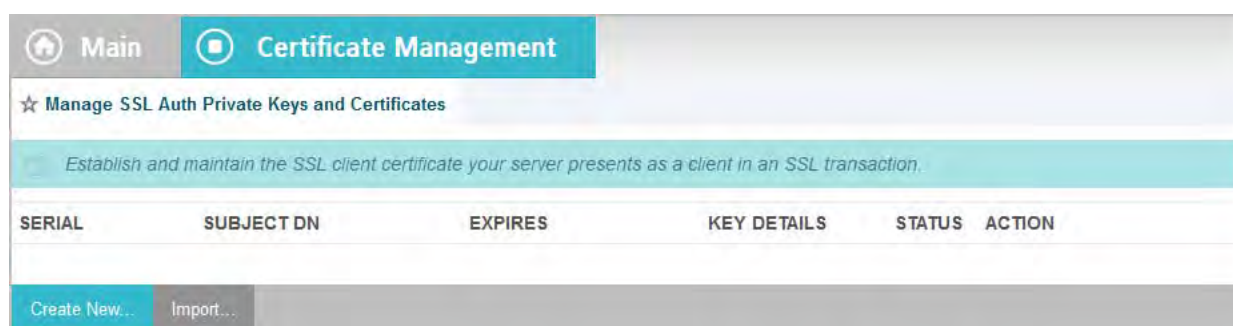
2. Click **Save**.

Create Certificate Field Descriptions

Field	Description
Common Name	The common name (CN) identifying the certificate.
Organization	The organization (O) or company name creating the certificate.
Organizational Unit	The specific unit within the organization (OU).
City	The city or other primary location (L) where the company operates.
State	The state (ST) or other political unit encompassing the location.
Country	The country (C) where the company is based.
Validity (days)	The time during which the certificate is valid.
Key Algorithm (drop-down menu)	A mathematical formula used to generate a key. PingFederate uses either of two algorithms, RSA or EC.
Key Size (bits)	The number of bits used in the key. (RSA-1024, 2048 and 4096, EC-256, 384 and 521)
Signature Algorithm (drop-down menu)	The signing algorithm of the certificate. (RSA-SHA256, SHA384 and SHA512, EC-ECDsa SHA256, SHA384 and SHA512)

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

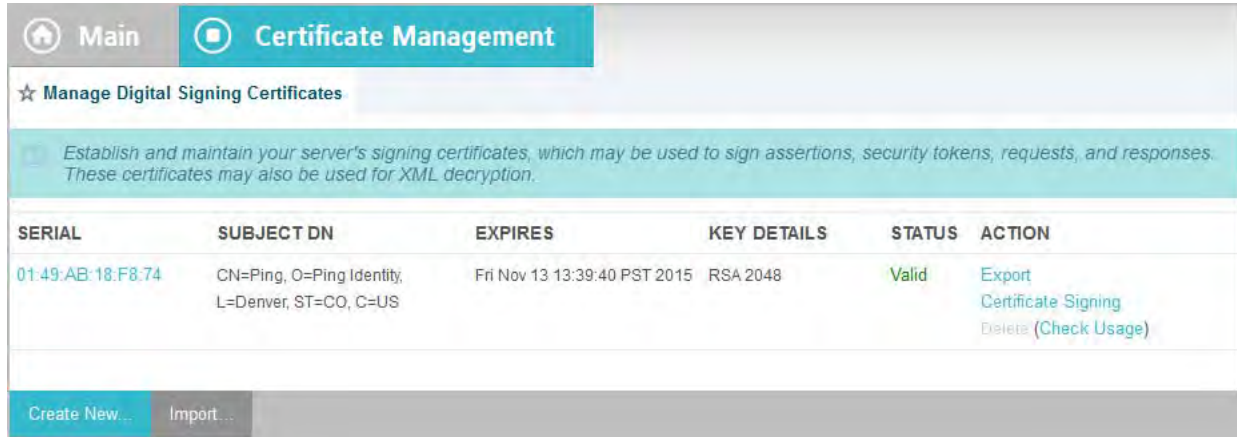
Field	Description
Common Name	The common name (CN) identifying the certificate.
Organization	The organization (O) or company name creating the certificate.
Organizational Unit	The specific unit within the organization (OU).
City	The city or other primary location (L) where the company operates.
State	The state (ST) or other political unit encompassing the location.
Country	The country (C) where the company is based.
Validity (days)	The time during which the certificate is valid.
Key Algorithm (drop-down menu)	A mathematical formula used to generate a key. PingFederate uses either of two algorithms, RSA or EC.
Key Size (bits)	The number of bits used in the key. (RSA-1024, 2048 and 4096, EC-256, 384 and 521)
Signature Algorithm (drop-down menu)	The signing algorithm of the certificate. (RSA-SHA256, SHA384 and SHA512, EC-ECDSA SHA256, SHA384 and SHA512)

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 31).



Caution: For best security, we recommend using separate certificates for signing and decryption.



The screenshot shows the 'Certificate Management' section of a web interface. At the top, there are navigation tabs for 'Main' and 'Certificate Management'. Below the tabs is a heading '★ Manage Digital Signing Certificates' and a descriptive text: 'Establish and maintain your server's signing certificates, which may be used to sign assertions, security tokens, requests, and responses. These certificates may also be used for XML decryption.' Below this is a table with columns: SERIAL, SUBJECT DN, EXPIRES, KEY DETAILS, STATUS, and ACTION. One certificate is listed with serial '01:49:AB:18:F8:74', subject 'CN=Ping, O=Ping Identity, L=Denver, ST=CO, C=US', expires 'Fri Nov 13 13:39:40 PST 2015', key 'RSA 2048', and status 'Valid'. The action column contains links for 'Export', 'Certificate Signing', and 'Delete (Check Usage)'. At the bottom of the interface are buttons for 'Create New...' and 'Import...'.

SERIAL	SUBJECT DN	EXPIRES	KEY DETAILS	STATUS	ACTION
01:49:AB:18:F8:74	CN=Ping, O=Ping Identity, L=Denver, ST=CO, C=US	Fri Nov 13 13:39:40 PST 2015	RSA 2048	Valid	Export Certificate Signing Delete (Check Usage)

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**.
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 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

Field	Description
Common Name	The common name (CN) identifying the certificate.
Organization	The organization (O) or company name creating the certificate.
Organizational Unit	The specific unit within the organization (OU).
City	The city or other primary location (L) where the company operates.

Field	Description
State	The state (ST) or other political unit encompassing the location.
Country	The country (C) where the company is based.
Validity (days)	The time during which the certificate is valid.
Key Algorithm (drop-down menu)	A mathematical formula used to generate a key. PingFederate uses either of two algorithms, RSA or EC.
Key Size (bits)	The number of bits used in the key. (RSA-1024, 2048 and 4096, EC-256, 384 and 521)
Signature Algorithm (drop-down menu)	The signing algorithm of the certificate. (RSA-SHA256, SHA384 and SHA512, EC-ECDSA SHA256, SHA384 and SHA512)

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 28). 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 28).

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.

Home **Main** Certificate Revocation Checking

★ Manage Certificate Revocation

Specify the certificate validation mechanism. For OCSP-based validation, specify the settings.

Enable OCSP

Default OCSP Responder URL

Default OCSP Responder Signature Verification Certificate - SELECT - ▾

Do NOT allow Responder to use cached responses

Validity Interval

This Update Grace Period (min) *

Next Update Grace Period (min) *

Timeout

Responder Timeout (sec) *

Response Caching Interval (hrs) *

Error Handling

Certificate is Unknown Treat as Revoked ▾

OCSP Responder is Unavailable Treat as Valid ▾

OCSP Responder Returns Error Treat as Revoked ▾

Enable CRL Checking

Treat Unretrievable CRLs as Revoked

Next Retry on Resolution Failure (min) *

Next Retry on Next Update Expiration (min) *

Verify CRL Signature

Proxy Settings

Host

Port



Note: No configuration changes are necessary on this screen if OCSP is not required for your federation deployment and the CRL defaults are acceptable.

To reach this screen:

- ▶ On the Main Menu under Certificate Management, click **Certificate Revocation Checking**.

Field Descriptions (For OSCP Checking)

Field/Selection	Description
Enable OCSP	Turns on OCSP certificate-revocation checking.
Default OCSP Responder URL	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.

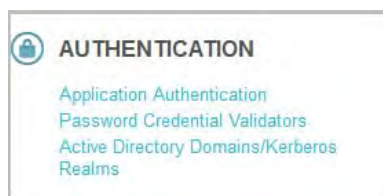
Field/Selection	Description
Default OCSP Responder Signature Verification Certificate	Certificate used to verify that the returned certificate status was sent from the Default OCSP Responder—required if the certificate is not included in the response (click Manage Certificates to import the verification certificate, as needed).
Do NOT allow Responder to use cached responses	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. Important: Making this selection may slow down OCSP response time for a request and will increase general processing overhead at the Responder site.
This Update Grace Period	For the response to be considered valid, the PingFederate server-clock time must correspond to the <code><thisUpdate></code> timestamp in the OCSP response, plus or minus the number of minutes set for this field (to compensate for clock variances).
Next Update Grace Period	If the response includes a <code><nextUpdate></code> 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.
Responder Timeout	The allowable response time before the OCSP Responder URL is considered unavailable and processing continues (see “OCSP Responder is Unavailable,” below).
Certificate is Unknown	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.
OCSP Responder is Unavailable	Indicates what action to take if the Responder cannot be reached.
OCSP Responder Returns Error	Indicates what action to take if the Responder returns an error.
Proxy Settings	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.

Field Descriptions (For CRL Checking)

Field/Selection	Description
Enable CRL Checking	Enables CRL revocation checking (the default). Note: 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.
Treat Unretrievable CRLs as Revoked	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.
Next Retry on Resolution Failure	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.
Next Retry on Next Update Expiration	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. Note: 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.
Verify CRL Signature	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 “Trusted Certificate Authorities” on page 222).
Proxy Settings	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.

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.



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 591).

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 571).

In addition, authentication is required to access PingFederate runtime data via JMX (see [“Runtime Monitoring Using JMX”](#) on page 107) 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 591).



Note: To help ensure network security, access to all of these services is deactivated when PingFederate is first installed.

SERVICE	ID	SHARED SECRET	CONFIRM SHARED SECRET	ACTION
Attribute Query	<input type="text"/>	<input type="text"/>	<input type="text"/>	Activate
JMX	<input type="text" value="jmx-user"/>	<input type="password"/>	<input type="password"/>	Deactivate
Connection Management	<input type="text" value="conn-mgmt-user"/>	<input type="password"/>	<input type="password"/>	Deactivate
SSO Directory Service	<input type="text"/>	<input type="text"/>	<input type="text"/>	Deactivate

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 543 and [“Configuring the HTML Form IdP Adapter”](#) on page 550) and for OAuth resource-owner grant configurations (see [“Grant Types”](#) on page 11 and [“Resource-Owner Credentials Mapping”](#) on page 195).

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.
- **PingOne directory Username/Password** – Validates credentials stored in PingOne directory.

★ **Manage Credential Validators**

Credential Validators are plug-ins used to verify username and password pairs in various contexts throughout the system. The actual application of a Validator instance must be configured in the appropriate context as needed (e.g., OAuth Resource Owner Credentials Mapping).

INSTANCE NAME	INSTANCE ID	TYPE	PARENT NAME	ACTION
Create New Instance...				

- ▶ 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**.)



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.

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](http://pingidentity.com/support-and-downloads) (pingidentity.com/support-and-downloads).

Field Descriptions

Field	Description
Instance Name	A descriptive name for the credentials-validator instance—for example, an identity management system name.
Instance ID	An internal identifier for the credentials-validator instance. Must be alphanumeric with no spaces.
Type	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.
Parent Instance (Optional)	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.

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 18). 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 239
- [“Configuring the RADIUS Credential Validator”](#) on page 240
- [“Configuring the PingOne Directory Credential Validator”](#) on page 242

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 550). The end-user messages are configurable for the associated HTML templates and may be localized (see [“Customizing User-Facing Screens”](#) on page 93).



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

Field Descriptions

Field	Description
LDAP Datastore	The LDAP data store configured in PingFederate. If you have not yet configured the server to communicate with the LDAP server you need, click Manage Data Stores . Note: If you are using Active Directory and want to support password changes (see “Configuring the HTML Form IdP Adapter” on page 550), 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 127).
Search Base	The location in the LDAP directory server from which the search begins.
Search Filter	Query used to produce the desired set of matching records.
Scope of Search	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.

To complete the LDAP validation:

- (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 change function or to alter end-user messages associated with that function (see ["HTML Form Adapter Configuration"](#) on page 549).

- Enter an expression, using wildcard asterisks (*), to match against messages returned from the LDAP server.
For example: *expired*
- Select a relevant error condition from the **Error** drop-down list.
- Click **Update** under Action.
- 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.)

- Select the LDAP Datastore and enter information into the required fields, as described under Field Descriptions above.

To edit an authentication error override:

- Click **Edit** next to the authentication error override.
- 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.

Main		Manage Credential Validator Instances		Create Credential Validator Instance	
Type	☆ Instance Configuration	Summary			
<p>Complete the configuration necessary for this Password Credential Validator to check username/password pairs. This configuration was designed into, and is specific to, the selected Credential Validator plug-in.</p>					
<p>This password credential validator provides a means of verifying credentials maintained by PingFederate.</p>					
<p>USERS (A table of valid usernames and passwords)</p>					
<p>USERNAME (An individual user name for authentication)</p> <input type="text" value="joe"/>	<p>PASSWORD (An individual password for authentication)</p> <input type="password" value="*****"/>	<p>CONFIRM PASSWORD (Must match password field)</p> <input type="password" value="*****"/>	<p>Action</p> <p>Edit Delete</p>		
<p>Add a new row to 'Users'</p>					
FIELD NAME	FIELD VALUE	DESCRIPTION			
<p>This plugin type has no individual configurable fields.</p>					



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

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 93).



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

Field Descriptions

Field	Description
Hostname	<p>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.</p> <p>PingFederate attempts to make a connection to each server in the order listed until a successful connection is obtained.</p> <p>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).</p>
Authentication Port	The UDP (User Datagram Protocol) port used to authenticate to the RADIUS server. The default value is 1812.
Authentication Protocol	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.
Shared Secret	The password shared between PingFederate and the RADIUS server used to encrypt passwords.



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:

- 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

Field	Description
NAS Identifier	(Required) The attribute identifying the NAS (Network Access Server) originating the request for access. The default value is <code>PingFederate</code> .
Timeout	(Required) The maximum number of milliseconds before a connection timeout to the RADIUS server.
Retry Count	(Required) Number of times to retry a failed connection before moving to the next host.

Configuring the PingOne Directory Credential Validator

The PingOne directory Username/Password Credential Validator verifies credentials stored in the PingOne directory.



Note: The PingOne directory Credential Validator requires a PingOne Enterprise or PingOne for Groups account. For more information, please contact sales@pingidentity.com.

Home Main Manage Credential Validator Instances **Create Credential Validator Instance**

Type **Instance Configuration** Extended Contract Summary

Complete the configuration necessary for this Password Credential Validator to check username/password pairs. This configuration was designed into, and is specific to, the selected Credential Validator plug-in.

PingOne Directory Password Credential Validator

FIELD NAME	FIELD VALUE	DESCRIPTION
CLIENT ID	<input type="text"/> *	A unique identifier PingFederate will use to identify itself to the PingOne directory API.
CLIENT SECRET	<input type="text"/> *	Used to authenticate against the PingOne directory API.

Show Advanced Fields



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

Field Descriptions

Field	Description
Client ID	The REST API Client ID of your PingOne account. Contact sales@pingidentity.com for more information.
Client Secret	The password for the REST API Client ID of your PingOne account. Contact sales@pingidentity.com for more information.
Note: To review the following settings, click Show Advanced Fields .	
PingOne URL	The PingOne directory API. The default value is <code>https://directory-api.pingone.com/api</code> .
OAuth Token Endpoint	The PingOne directory API OAuth 2.0 token endpoint. The default value is <code>/oauth/token</code> .
Reset Password URL	The relative path for password reset. The default value is <code>/directory/users/password-reset</code> .
SCIM User URL	The SCIM GET User endpoint. The default value is <code>/directory/user</code> .

To complete the configuration:

- ▶ Enter a value for each field as described above.

Extending Contracts for the Credential Validator

In some cases, you might want to extend contracts in the LDAP, PingOne directory or 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 195) or attributes in the access token (see “Token Attribute Contract Fulfillment” on page 206).

This capability allows the validator to return attribute values pertaining to the authenticated users from the LDAP server, the PingOne directory, or the RADIUS server.



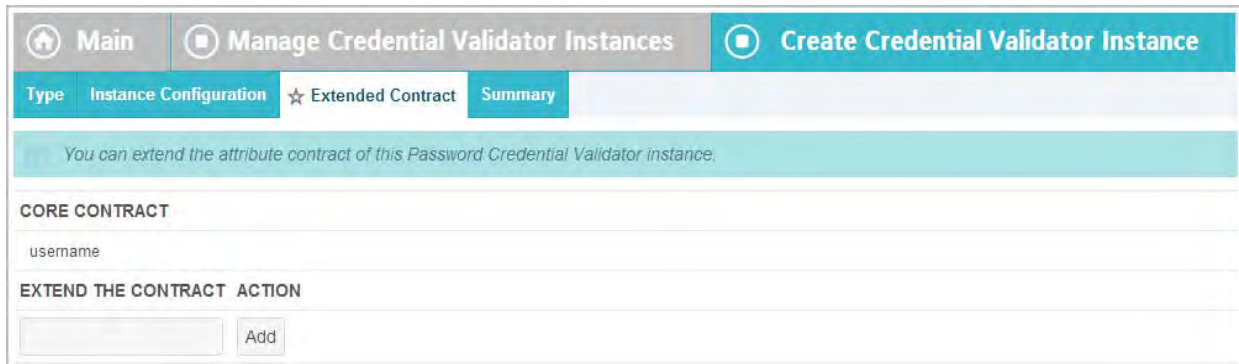
Tip: If you are configuring the HTML Form IdP Adapter with the LDAP credential validator, extend the contract of the adapter by the same attribute names in order for the credential validator to pass extended attribute values to the HTML Form IdP Adapter (see “Configuring the HTML Form IdP Adapter” on page 550).



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 550).

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 the contracts for a 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.

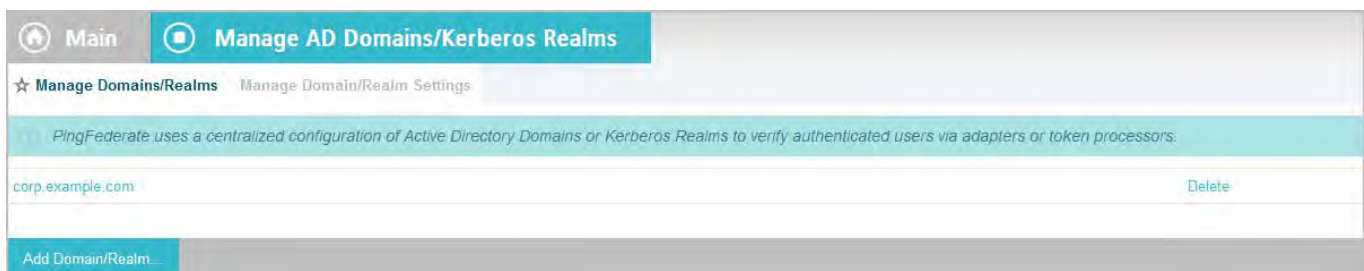
Using AD Domains and Kerberos Realms

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

Field	Description
Domain/Realm Name	The fully-qualified domain or realm name. For example: corporation.companydomain.com
Domain/Realm Username	The ID for the domain or realm account name.
Domain/Realm Password	The password for the domain or realm account.
Domain Controller/Key Distribution Center Host Names	(Optional) Specify the host name or IP address for the domain controller or KDC and click Add . For example: fn_dc1 If unspecified, PingFederate uses a DNS lookup.

Field	Description
Test Domain/Realm Connectivity (button)	<p>Tests access to the domain controller or KDC from the administrative-console server.</p> <p>When a connection to any of the configured controllers/KDCs is successful, the message <code>Test Successful</code> appears. Otherwise, the test returns error messages near the top of the screen.</p> <p>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.</p> <p>Tip: For help resolving connection issues, select the Debug Log Output checkbox on the Manage Domain/Realm Settings screen, and run the test again to view the debug output to the <code>PingFederate server.log</code> (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).</p>

Managing Domain or Realm Settings

Use this screen to change default security and logging settings for all configured AD Domains and Kerberos Realms.

Field Descriptions

Field	Description
Force TCP	<p>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.</p> <p>Note: If you choose this option, ensure that you restart PingFederate after saving the configuration.</p>

Field	Description
Debug Log Output	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 50.
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.

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 111).

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 38).



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 469.

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 32).

You can also deploy an SP connection to bridge a service provider to one or more identity providers through one or more connection mapping contracts (see [“Federation Hub”](#) on page 42 and [“Connection Mapping Contracts”](#) on page 162 for more information).

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 15).

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 475.

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 15).

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 534
- “[Configuring the HTTP Basic IdP Adapter](#)” on page 543
- “[Configuring the HTML Form IdP Adapter](#)” on page 550
- “[Configuring the Composite Adapter](#)” on page 557

INSTANCE NAME	INSTANCE ID	TYPE	PARENT NAME	ACTION
OpenToken	OTK1	OpenToken Adapter 2.5.1		Delete

- ▶ 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 15). Other adapters and integration kits are available from the [Ping Identity Web site](http://www.pingidentity.com/support-and-downloads) (www.pingidentity.com/support-and-downloads).

The screenshot shows the 'Create Adapter Instance' form. At the top, there are navigation tabs: 'Main', 'Manage IdP Adapter Instances', and 'Create Adapter Instance' (which is active). Below the tabs, there are breadcrumb links: '☆ Type', 'IdP Adapter', 'Adapter Attributes', and 'Summary'. A teal banner contains the instruction: 'Enter an Adapter Instance Name and Id, select the Adapter Type, and a parent if applicable. The Adapter Type is limited to the adapters currently installed on your server.' The form fields are: 'Instance Name' (text input with an asterisk), 'Instance Id' (text input with an asterisk), 'Type' (dropdown menu with '- SELECT -' and a link to 'Visit Pingidentity.com for additional types'), and 'Parent Instance' (dropdown menu with 'None').

Field Descriptions

Field	Description
Instance Name	A descriptive name for the adapter instance—for example, an identity management system name.
Instance ID	An internal identifier for the adapter instance. Must be alphanumeric with no spaces.
Type	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.
Parent Instance (Optional)	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.

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 15).

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 18). 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.



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 15). For information about configuring the adapters packaged with PingFederate, see “[OpenToken Adapter Configuration](#)” on page 533, “[Configuring the HTTP Basic IdP Adapter](#)” on page 543, “[Configuring the HTML Form IdP Adapter](#)” on page 550, or “[Configuring the Composite Adapter](#)” on page 557.

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 15).

- For information about configuring the OpenToken Adapter, see “[OpenToken Adapter Configuration](#)” on page 533.
- For information about configuring the HTTP Basic IdP Adapter, see “[Configuring the HTTP Basic IdP Adapter](#)” on page 543.
- For information about configuring the HTML Form IdP Adapter, see “[Configuring the HTML Form IdP Adapter](#)” on page 550.
- For information about configuring the Composite Adapter, see “[Configuring the Composite Adapter](#)” on page 557.
- 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 534.

Main Manage IdP Adapter Instances Create Adapter Instance		
Type	IdP Adapter	★ Actions
Extended Contract Adapter Attributes Summary		
These are the actions that this adapter type can perform.		
ACTION NAME	ACTION DESCRIPTION	ACTION INVOCATION LINK
Download	Download the configuration file for the agent.	Invoke Download

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 533).



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 557).

Main Manage IdP Adapter Instances Create Adapter Instance		
Type	IdP Adapter	★ Actions
Extended Contract Adapter Attributes Summary		
This adapter type supports the creation of an Extended Adapter Contract after initial deployment of the adapter instance. This Adapter Contract may be used to fulfill the Attribute Contract, look up additional attributes from a local data store, or create a persistent name identifier which uniquely identifies the user passed to your SP partners.		
CORE CONTRACT		
subject		
EXTEND THE CONTRACT ACTION		
<input type="text"/>	<input type="button" value="Add"/>	

To reach this screen:



Note: If this is a child instance, select the override checkbox to modify the configuration.

1. Click **Adapters** on the Main Menu.
2. Click an Instance Name.
3. Click **Extended Contract** (if available).

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 19).

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 25).

ATTRIBUTE	PSEUDONYM	MASK LOG VALUES
subject	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Mask all OGNL-expression generated log values



Note: If this is a child instance, select the override checkbox to modify the configuration.



Tip: If the IdP Adapter supports the creation of an Extended Adapter Contract (see the previous section, “[Extending an Adapter Contract](#)”), then the Override Attributes checkbox in this screen reflects the status of the override option in the Extended Contract screen.

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 613).

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 IdP 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 Authentication Selectors

Authentication selectors provide a plug-in capability for PingFederate to choose among configured authentication sources (IdP adapter instances or IdP connections) for any SSO request and some OAuth flows that involve the browser. PingFederate chooses authentication sources based on criteria you define in a configured authentication selector or series of selectors. This optional feature provides for global and flexible authentication-source usage across SP connections.



Important: Before configuring authentication selectors, it is best to configure the authentication sources and to associate them with the applicable SP connections (see [“Authentication Source Mapping”](#) on page 287).

The configuration of plug-in selectors packaged in this PingFederate release—see [“Bundled Authentication Selectors”](#) on page 16—is described in the following sections. For information on the configuration of other plug-ins, please consult the associated documentation.

To reach this screen:

- ▶ Click Authentication Selection under Application Integration Settings in IdP Configuration.

To configure an instance of an authentication 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 authentication selector to configure. Many selector plug-ins are bundled with PingFederate (see “[Bundled Authentication Selectors](#)” on page 16). Other plug-ins may be added periodically, available from [Ping Identity](http://pingidentity.com/support-and-downloads) (pingidentity.com/support-and-downloads).

Field Descriptions

Field	Description
Instance Name	A descriptive name for the selector instance.
Instance ID	An internal identifier for the selector instance—must be alphanumeric with no spaces.
Type	A list of deployed selector types available for creating an instance for the server.

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 authentication-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 Authentication Selector](#)” on page 257
- “[Configuring the Cluster Node Authentication Selector](#)” on page 257

- “Configuring the Connection Set Authentication Selector” on page 258
- “Configuring the HTTP Header Authentication Selector” on page 259
- “Configuring the OAuth Scope Authentication Selector” on page 260
- “Configuring the Requested AuthN Context Authentication Selector” on page 261

Configuring the CIDR Authentication Selector

This authentication selector determines the authentication source 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 Authentication 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 authentication sources configured as a result of this authentication selector must have their attribute contract extended with the Result Attribute Name value in order to use its value to fulfill an attribute contract or for issuance criteria.



Note: Result Attribute Name (if specified) is provided to the chosen authentication source only when the CIDR authentication selector is the last test. For more information, see “[Mapping Selector Results to Authentication Sources](#)” on page 262.

Configuring the Cluster Node Authentication Selector

This authentication selector enables PingFederate to choose configured authentication sources 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 Authentication 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 authentication sources (see “[Mapping Selector Results to Authentication Sources](#)” on page 262). These results are used as the criteria for authentication 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 authentication selection.

Configuring the Connection Set Authentication Selector

This authentication selector enables PingFederate to choose configured authentication sources 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 authentication 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.

- 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 Authentication Selector

This authentication selector enables PingFederate to choose configured authentication sources based on a match found in a specified HTTP header. For example, use an instance of this selector to choose an authentication source based on the end user's browser identified by the User-Agent HTTP header.

Complete the configuration needed for this Selector Instance.

This authentication selector provides a means of choosing authentication sources at runtime based on HTTP headers.

RESULTS (A table of expressions to match against the values for the given header name. If any expression matches, the result value is Yes.)

MATCH EXPRESSION (The expression matched against the specified header.)	Action
MSIE	Move down Edit Delete
rv:11	Move up Edit Delete

Add a new row to Results

FIELD NAME	FIELD VALUE	DESCRIPTION
HEADER NAME	<input type="text"/>	HTTP header to inspect for a match.

To configure the selector:

- Under Action, click **Add a new row to 'Results'**.
- 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.

Browser	Expression
Chrome	*Chrome*
Firefox	*Firefox*
iPhone	*iPhone*
iPad	*iPad*
Internet Explorer	*MISE*
Safari	*Safari*

For information about Internet Explorer 11 (or higher), see [User-agent string changes](https://msdn.microsoft.com/en-us/library/ie/hh869301(v=vs.85).aspx) from Microsoft ([msdn.microsoft.com/en-us/library/ie/hh869301\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/ie/hh869301(v=vs.85).aspx)).

- 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 Authentication Selector

This authentication selector enables PingFederate to choose configured authentication sources 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 169).

🏠 Main
📄 Manage Authentication Selector Instances
➤ Create Authentication Selector Instance

Type
☆ Authentication Selector
Summary

Complete the configuration needed for this Selector Instance.

This authentication selector provides a means of choosing authentication sources at runtime based on the OAuth scope request received by the Authorization Endpoint. Select all scopes required to trigger a “Yes” result from this selector. Create multiple instances to handle boolean ‘OR’ cases.

FIELD NAME	FIELD VALUE	DESCRIPTION
ADDRESS	<input type="checkbox"/>	Address access
ADMIN	<input type="checkbox"/>	Admin permissions
EDIT	<input type="checkbox"/>	Edit permissions
EMAIL	<input type="checkbox"/>	Email Address access
OPENID	<input type="checkbox"/>	OpenID Connect login
PHONE	<input type="checkbox"/>	Phone Number access
PROFILE	<input type="checkbox"/>	Profile access
QUICKCONTACT	<input checked="" type="checkbox"/>	Quick contact info (Group)

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 do not match this authentication 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 Authentication Sources”](#) on page 262).

Configuring the Requested AuthN Context Authentication Selector

This authentication selector enables PingFederate to choose configured authentication sources based on the [authentication context](#) requested by an SP partner. The Authentication 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 261.)



Note: This authentication 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.

Main		Manage Authentication Selector Instances		Create Authentication Selector Instance	
Type	★ Authentication Selector	Selector Result Values	Summary		
Complete the configuration needed for this Selector Instance.					
This authentication selector selects an authentication source based on the authentication context requested by an SP, for SP-initiated SSO or through OpenID Connect. SAML-specified contexts, or any ad-hoc context agreed upon between the IdP and SP partners, are specified on the Selector Result Values screen.					
FIELD NAME	FIELD VALUE	DESCRIPTION			
ADD OR UPDATE AUTHN CONTEXT ATTRIBUTE	<input checked="" type="checkbox"/>	Indicates (when specified) if the AuthN Context attribute value will be updated with the authentication selector result value.			

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.

Authentication sources are mapped to the results of selector instances in a subsequent screen, Map Results to Authentication Sources.



Note: The authentication context is modified only when the Requested AuthnN Context authentication selector is the last test. For more information, see [“Mapping Selector Results to Authentication Sources”](#) on page 262.

Defining AuthN Context Results

On the Selector Result Values screen, list the authentication contexts to be associated with specific authentication sources (see [“Mapping Selector Results to Authentication Sources”](#) on page 262). These results are also used as the criteria for authentication selection.

Main		Manage Authentication Selector Instances		Create Authentication Selector Instance	
Type	Authentication Selector	★ Selector Result Values	Summary		
Specify expected result values. Each result value will be mapped to an appropriate Authentication Source.					
RESULT VALUES	ACTION				
<input type="text"/>	<input type="button" value="Add"/>				

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 authentication selection.

Finishing the Selector-Instance Configuration

To complete the selector-instance configuration, click **Done** on the Summary screen.

- ▶ On the Manage Authentication Selector Instances screen, add more instances or click **Next** to complete the selector configuration (see the next section, “[Mapping Selector Results to Authentication Sources](#)”).



Caution: If you are yet not ready to map results, be sure to click **Save** on the Manage Authentication Selector Instances screen. Selector-instance configurations are not retained until you click **Save** either on this screen or on the Map Results to Authentication Sources screen.

Mapping Selector Results to Authentication Sources

An administrator can map the results of selector instances either to authentication sources 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 authentication-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 Authentication Source to Certain Virtual Server IDs](#)” on page 291), PingFederate falls back on the Default Authentication Sources (if any) or selects among the available authentication sources configured for the applicable SP connection (see [Step 2](#) and [Step 7](#)).

Change the order as needed using the arrows in the left column.

Main Manage Authentication Selector Instances

Manage Authentication Selector Instances ★ Map Results to Authentication Sources

Enable Authentication Selector Instances to be applied across connections as authentication policy during SSO processing. Multiple Selector Instances are applied in the specified order. Map each selector result value to an IdP Authentication Source or to another Authentication Selector for evaluation of additional criteria.

Enable Authentication Selection
 Fail If No Selection

SELECTOR INSTANCE NAME	SELECTOR RESULT VALUES	INSTANCE	SELECTOR RESULT VALUES	INSTANCE
▼ ACME IdP Company	No	--None--		
	Yes	IdP1 to fedhub - (IdP Connection)		
▲ Internal Applications	No	--None--		
	Yes	Internet Explorer - (Selector)	No	IdpAdapter - (Adapter)
			Yes	IdpOpenToken - (Adapter)
		--None--		
DEFAULT AUTHENTICATION SOURCES				
--None--				

To complete this configuration:

1. Ensure the checkbox Enable Authentication Selection is selected.



Tip: This checkbox allows an administrator to turn authentication selection on and off and is primarily useful during deployment testing. If this checkbox is *not* selected, all authentication-selection mappings are deactivated.

2. (Optional) Select the checkbox Fail If No Selection.

If no authentication sources match either the selector-result mappings or any Default Authentication Sources listed at the bottom of the screen (see [Step 7](#)), PingFederate selects among the authentication sources 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 authentication sources 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 authentication source 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 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.



Tip: There is no limit to the depth of the decision-tree processing this configuration makes possible; use the browser's horizontal scroll bar, if needed, to view the rightmost columns.

- Repeat [Step 3](#) through [Step 5](#) for each Selector Instance Name that needs mapping.
- (Optional) Under Default Authentication Sources, choose an authentication source from the drop-down list.

Repeat this step to choose additional default instances, if needed.

If no authentication sources match any of the selector mappings at runtime, the authentication sources listed in this column are tried in order. The first authentication source that is mapped in the target SP connection configuration is the one used.

- 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 use the PingFederate localization feature, see [“Localization”](#) on page 98. 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 563), 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 93).

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 15).



Note: For specific parameters required or allowed for Application Endpoints, see [“IdP Endpoints”](#) on page 563.

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 576).

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 111). These endpoints are built into PingFederate and cannot be changed.

PingFederate provides a favorite icon for all Protocol Endpoints. For more information, see [“Customizing the Favicon for Application and Protocol Endpoints”](#) on page 101.

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 63).

The table below describes each endpoint:

Table 21: PingFederate IdP Endpoints

Service	URL and Description
Single Logout Service (SAML 2.0)	/idp/SLO.saml2 The URL that receives and processes logout requests and responses.
Single Sign-on Service (SAML 2.0)	/idp/SSO.saml2 The SAML 2.0 implementation URL that receives authentication requests for processing.
Artifact Resolution Service (SAML 2.0)	/idp/ARS.ssaml2 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.)
Attribute Query Service (SAML 2.0)	/idp/attrsvc.ssaml2 The SAML implementation that receives and processes attribute requests. (See “ Important ” footnote in this table.)
Metadata Service	/ The default endpoint (empty path) from which partners can retrieve Auto-Connect metadata (see “ Using Auto-Connect ” on page 32).
Single Sign-on Service (SAML 1.x)	/idp/isx.saml1 The SAML 1.x implementation of IdP intersite transfer service (ISX) to which clients are redirected for SSO requests.
Artifact Resolution Service (SAML 1.x)	/idp/soap.ssaml1 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.)
Single Sign-on Service (WS-Federation)	/idp/prp.wsf The WS-Federation implementation URL that receives and processes security-token requests and SLO messages.

Table 21: PingFederate IdP Endpoints (Continued)

Service	URL and Description
WS-Trust STS (two endpoints)	<p data-bbox="919 302 1089 327"><code>/idp/sts.wst</code></p> <p data-bbox="919 338 1409 485">The SOAP endpoint that receives and processes security-token requests from STS clients (Web Service Clients at the IdP site) to be exchanged for a SAML token based on the configured SP connection.</p> <p data-bbox="919 501 1073 527"><code>/pf/sts.wst</code></p> <p data-bbox="919 537 1409 684">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 “Token Exchange Mapping” on page 155).</p> <p data-bbox="919 701 1409 911">Note: If multiple token-processor instances of the same type are configured for the same connection or token-to-token mapping, a query parameter, <code>TokenProcessorId</code>, must be added to either of these endpoints—see “Configuring Token Processors” on page 475.</p> <p data-bbox="919 928 1354 982">(See also “Important” footnote in this table.)</p>
<p data-bbox="537 1005 1390 1119">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—<code>*.ssaml*</code> and <code>*.wst</code> (see “Changing Configuration Parameters” on page 80).</p>	



Note: For any connection using multiple virtual server IDs (see [“Multiple Virtual Server IDs”](#) on page 39), each virtual server ID has its own set of protocol endpoints. Use Export Metadata on Main Menu to export a connection metadata for your partner. For more information, see [“Exporting Metadata”](#) on page 63.

Managing SP Connections

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 276. Then turn to [“WS-Trust STS Configuration”](#) on page 469.

These settings include:

- 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 37). Your federation partner must supply some connection settings and other information (see “[Configuration Data Exchange](#)” on page 40).



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 356).

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 122).

Accessing Connections

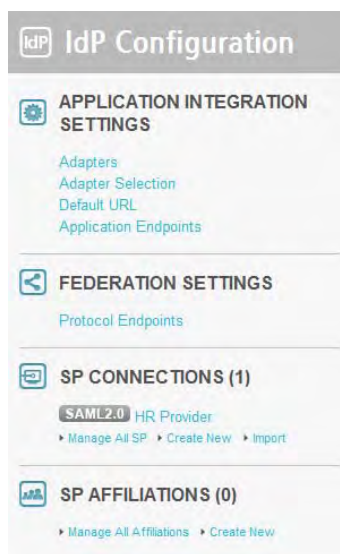
You can create, modify or import connections directly via the Main Menu. Note that the menu displays the four most-recently modified connections. To view a list of all SP connections, click the **Manage All SP** link.

Via the Main Menu

From the Main Menu under IdP Configuration, you can configure a new connection, modify an existing connection, import a connection, or view connections.



Tip: To copy, delete or export connections, as well as to recover connection drafts, click **Manage All SP** (see “[Via the Manage Connections Screen](#)” on page 269).



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 [“Via the Manage Connections Screen”](#) on page 269).

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 269).

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.

To import a connection:

1. Click **Import**.
2. In the Import Connection screen, browse to a connection XML file.
3. (Optional) Select the Allow Update checkbox. When selected, if the connection already exists, it will be overwritten.
4. Click **Import** and **Done**.

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 or import individual connection configurations.



Note: The connection export function results in an XML file that you can modify and import into the same PingFederate server or another PingFederate server acting in the same federation role (IdP or SP) at your site. You can import connections in this screen. Alternatively, you can use the Connection Management Service to complete the task (see [“Importing Connections”](#) on page 593). Finally, you also have the option to automate this process (see [“Automating Configuration Migration”](#) on page 84).

- Export metadata about a connection to expedite your partner’s corresponding configuration (see [“Exporting Metadata”](#) on page 63).

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 53).



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 272.

★ Manage Connections

On this screen you can manage connections to your partner SPs. Use the drop-downs to filter the connection list. You can also override the logging mode for all SP connections by specifying a single, global logging mode.

CONNECTION NAME ▼	CONNECTION ID ▼	PROTOCOL ▼	STATUS ▼	ACTION
<input checked="" type="checkbox"/> HR Provider	HRProvider	SAML 2.0	Inactive	Delete Copy Export Connection Export Metadata

Create Connection... Import Connection

Logging Mode Override

Off
 On

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.
2. Update the Connection ID (of the partner) and the Connection Name in the General Info screen (see [“General Information”](#) on page 276).

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 35).

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.

To import a connection:

1. Click **Import**.
2. In the Import Connection screen, browse to a connection XML file.
3. (Optional) Select the Allow Update checkbox. When selected, if the connection already exists, it will be overwritten.
4. Click **Import** and **Done**.

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 63).
2. Complete the steps remaining in the Export Metadata screen flow (starting at [Step 4](#) under [“To export connection metadata:”](#) on page 64).

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.



Note: The Virtual ID displays the default virtual server ID of the connection, if any (see [“General Information”](#) on page 276).

To filter the list by Protocol and/or Status:

- ▶ Select a filter criterion from either or both of the drop-down lists.

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.

To import a connection:

1. Click **Import**.
2. In the Import Connection screen, browse to a connection XML file.
3. (Optional) Select the Allow Update checkbox. When selected, if the connection already exists, it will be overwritten.
4. Click **Import** and **Done**.

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 116).

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. You may, however, still edit the connection by clicking its name.

In addition, Action links are disabled (except for **Delete**, if the connection is Inactive and/or In Use) until the connection is validated.

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 292). 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 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 35).



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.

The screenshot shows the 'Connection Template' screen in the 'SP Connection' section. The 'Connection Template' tab is active. A message states: 'PingFederate provides quick-configuration templates, available separately with SaaS Connectors, for specific Service Providers. If applicable, please select a template for this connection, otherwise, continue to the next screen for more options.' Two radio buttons are present: 'Do not use a template for this connection' (selected) and 'Use a template for this connection'.

► To configure a connection without a template, click **Next**.

The screenshot shows the 'Connection Template' screen with 'Use a template for this connection' selected. The 'Connection Template' dropdown menu is set to 'Google Apps'. The 'Google Domain' text input field contains 'example.com'. There is a checkbox for 'Use a domain specific issuer' which is currently unchecked.

► To use a template, select that option, then choose the template and enter additional information as required.



Note: Once you click **Next**, you cannot return to this screen and make a different selection. If you intended to use a different template or no template, you must create a new connection.

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.

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 111).



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 111).

- ▶ (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 273).

The screenshot shows the 'SP Connection' configuration interface. The 'Connection Options' tab is active. A message reads: 'Please select options that apply to this connection:'. Below this, there are three checkboxes: 'Browser SSO' (checked), 'IdP Discovery' (unchecked), and 'Attribute Query' (unchecked).

- ▶ 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 139).

- ▶ 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), PingFederate 7.3 uses this information to populate the Base URL field (see [“General Information”](#) on page 276). 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 53).

The screenshot displays the 'General Info' tab within the 'SP Connection' configuration interface. The interface includes a navigation bar with 'Main' and 'SP Connection' tabs, and a sub-navigation bar with 'Connection Type', 'Connection Options', 'Import Metadata', 'General Info', 'Browser SSO', 'Credentials', and 'Activation & Summary'. A teal informational banner at the top explains the fields. Below the banner are several input fields: 'Partner's Entity ID (Connection ID)', 'Connection Name', 'Virtual Server IDs' (with an 'Add' button), 'Base URL', 'Company', 'Contact Name', 'Contact Number', 'Contact Email', 'Application Name', and 'Application Icon URL'. At the bottom, the 'Logging Mode' is set to 'Standard' via radio buttons, with other options being 'None', 'Enhanced', and 'Full'.

Field Descriptions

Field	Description
Partner's Entity ID/ Audience/ Partner's Realm (Connection ID)	(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.
Connection Name	(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.
Virtual Server IDs	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).
Base URL	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.
Company	The name of the partner company to which you are connecting.
Contact Name	The contact person at the partner company.
Contact Number	The phone number of the contact person at the partner company.
Contact Email	The email address for the contact person at the partner company.
Application Name	The name of the application, accessible through the IdP Adapter interface (<code>IdpAuthenticationAdapterV2</code>) in the PingFederate Java SDK. (For more information about the SDK, see SDK Developer's Guide .)
Application Icon URL	The URL of the application icon, accessible through the IdP Adapter interface (<code>IdpAuthenticationAdapterV2</code>) in the PingFederate Java SDK. (For more information about the SDK, see SDK Developer's Guide .)
Logging Mode	The level of transaction logging applicable for this connection (see).

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).

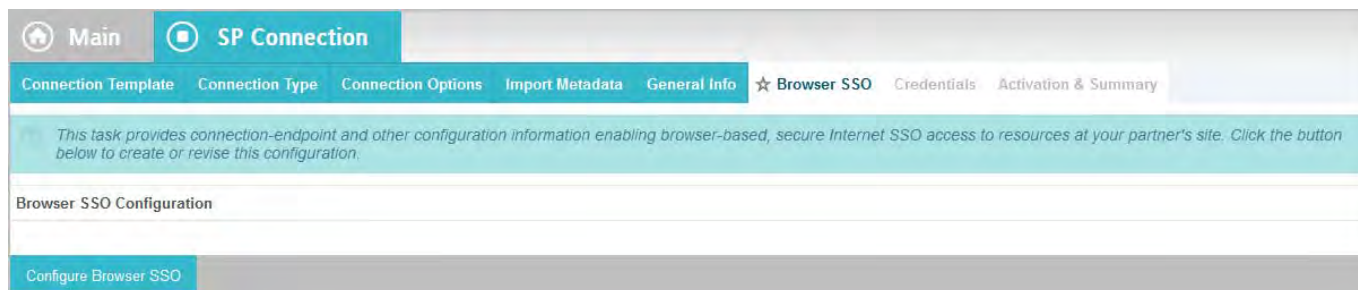
Note that the values in Virtual Server IDs identify your own federation deployment for this connection only and override the ID you specified under Server Settings (see “[Federation Server Identification](#)” on page 38).

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 330). 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 275).



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 280
- “[Setting an Assertion Lifetime](#)” on page 282
- “[Assertion Creation](#)” on page 283
 - “[Choosing an Identity Mapping Method](#)” on page 283
 - “[Creating an Attribute Contract](#)” on page 285
 - “[Authentication Source Mapping](#)” on page 287
- “[Configuring Protocol Settings](#)” on page 313
 - “[Setting Assertion Consumer Service URLs \(SAML\)](#)” on page 314
 - “[Specifying SLO Service URLs \(SAML 2.0\)](#)” on page 317
 - “[Choosing Allowable SAML Bindings \(SAML 2.0\)](#)” on page 318
 - “[Setting an Artifact Lifetime \(SAML\)](#)” on page 319
 - “[Specifying Artifact Resolver Locations \(SAML 2.0\)](#)” on page 319
 - “[Defining Signature Policy](#)” on page 320
 - “[Configuring XML Encryption Policy \(SAML 2.0\)](#)” on page 321

WS-Federation SSO Configuration Steps

- “[Setting an Assertion Lifetime](#)” on page 282
- “[Assertion Creation](#)” on page 283
 - “[Choosing an Identity Mapping Method](#)” on page 283
 - “[Creating an Attribute Contract](#)” on page 285

- “Authentication Source Mapping” on page 287
- “Configuring Protocol Settings” on page 313
 - “Defining a Service URL (WS-Federation)” on page 316

SAML 1.x SSO Configuration Steps

- “Setting an Assertion Lifetime” on page 282
- “Assertion Creation” on page 283
 - “Choosing an Identity Mapping Method” on page 283
 - “Creating an Attribute Contract” on page 285
 - “Authentication Source Mapping” on page 287
- “Configuring Protocol Settings” on page 313
 - “Setting Assertion Consumer Service URLs (SAML)” on page 314
 - “Setting a Default Target URL (SAML 1.x)” on page 315
 - “Setting an Artifact Lifetime (SAML)” on page 319
 - “Defining Signature Policy” on page 320

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 37). 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*.

The screenshot shows the configuration interface for SAML Profiles. At the top, there are navigation tabs: Main, SP Connection, and Browser SSO. Below the tabs, there are sub-tabs: SAML Profiles, Assertion Lifetime, Assertion Creation, Protocol Settings, and Summary. A descriptive text box states: "A SAML Profile defines what kind of messages may be exchanged between an Identity Provider and a Service Provider, and how the messages are transported (bindings). As an IdP, you configure this information for your SP connection." Below this, there are two columns of profile selection options:

Single Sign-On (SSO) Profiles	Single Logout (SLO) Profiles
<input checked="" type="checkbox"/> IdP-Initiated SSO	<input checked="" type="checkbox"/> IdP-Initiated SLO
<input checked="" type="checkbox"/> SP-Initiated SSO	<input checked="" type="checkbox"/> SP-Initiated SLO



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 27).

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.

Field Descriptions

Field	Description
Minutes Before	The amount of time before the assertion was issued during which it is to be considered valid.
Minutes After	The amount of time after the assertion was issued during which it is to be considered valid.

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.

If you are a federation hub, bridging a service provider to one or more identity providers, you must associate one or more connection mapping contracts to the SP connection (see [“Federation Hub”](#) on page 42 and [“Connection Mapping Contracts”](#) on page 162 for more information).

For both scenarios, the 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 285).
- Configuring instances of one or more authentication sources (see [“Authentication Source Mapping”](#) on page 287) and specifying how they are used to fulfill the contract.

The screenshot shows the configuration interface for Assertion Creation. At the top, there are navigation tabs: Main, SP Connection, and Browser SSO. Below these are sub-tabs: SAML Profiles, Assertion Lifetime, Assertion Creation (selected), Protocol Settings, and Summary. A teal banner contains the text: "This task provides the configuration for creating SAML assertions to enable SSO access to resources at your SP partner's site." Below this is a table titled "Assertion Configuration" with the following rows:

Assertion Configuration	
Identity Mapping	Standard
Attribute Contract	SAML_SUBJECT
Adapter Instances	0
Connection Contract Mappings	0

At the bottom of the interface, there is a button labeled "Configure Assertion Creation".

- Click **Configure Assertion Creation** to continue.

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 18). 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 37).

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 285.

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](#).

Standard: Send the SP a known attribute value as the name identifier. The SP will often use account mapping to identify the user locally.

Pseudonym: Send the SP a unique, opaque name identifier that preserves user privacy. The identifier cannot be traced back to the user's identity at this IdP and may be used by the SP to make a persistent association between the user and a specific local account. The SP will often use account linking to identify the user locally.

 Include attributes in addition to the pseudonym.

Transient: Send the SP an opaque, temporary value as the name identifier.

 Include attributes in addition to the transient identifier.

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 20).



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 254).

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 19).

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 311).

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 21). You identify these attributes on this screen.

If you are sending a “standard” name identifier (see “[Choosing an Identity Mapping Method](#)” on page 283), 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 304).



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 283).

Main		SP Connection		Browser SSO		Assertion Creation	
Identity Mapping		★ Attribute Contract		Authentication Source Mapping		Summary	
An Attribute Contract is a set of user attributes that this server will send in the assertion.							
ATTRIBUTE CONTRACT		SUBJECT NAME FORMAT					
SAML_SUBJECT		urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified *					
EXTEND THE CONTRACT		ATTRIBUTE NAME FORMAT				ACTION	
		urn:oasis:names:tc:SAML:2.0:attname-format:basic				Add	

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 283).



Note: For WS-Federation connections, the Subject Name Format is determined by the previous screen (see [“WS-Federation Name ID Selections”](#) on page 285).

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 304).

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 22).



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 22).



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 283).

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.

Authentication Source Mapping

IdP adapters are responsible for handling user authentication as part of an SSO operation (see [“SSO Integration Kits and Adapters”](#) on page 15). A configured and deployed adapter in PingFederate is known as an adapter instance. The same instance may be mapped by multiple connections.

You may also deploy an SP connection to bridge a service provider to one or more identity providers.



Tip: In this scenario, PingFederate is a federation hub for both parties. (See [“Federation Hub”](#) on page 42 and [“Bridging multiple IdPs to an SP”](#) on page 44 for more information.)

PingFederate uses connection mapping contracts to associate this SP connection with the applicable IdP connections to the identity providers (see [“Connection Mapping Contracts”](#) on page 162).

Each connection mapping contract has its own set of attributes which you map values to the assertions.

Map one or more IdP adapter instances or connection mapping contracts 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 authentication sources (IdP adapter instances or connection mapping contracts) are restricted to certain virtual server IDs, the allowed IDs are displayed under the Virtual Server IDs column.

Regardless of how many authentication sources are mapped in an SP connection, PingFederate uses only one instance to authenticate a user. Because each instance may return different user attributes, each authentication source must define how the [attribute contract](#) is fulfilled; you must map attributes from each authentication source—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 Authentication Source Mapping screen, where you choose to map instances of IdP adapters or connection mapping contracts. 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 250. To review or create connection mapping contracts, see [“Managing Contracts”](#) on page 162.

To modify an existing authentication source:

- ▶ Click its Name link.

To begin configuring an Adapter Instance for this connection:

- ▶ Click **Map New Adapter Instance**.

To begin configuring a connection mapping contract for this connection:

- ▶ Click **Map New Connection Contract Mapping**.

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 **Authentication Source Mapping** on the Summary screen.

Selecting an Authentication Source

Use this screen to associate an IdP adapter instance or a connection mapping contract with an SP connection.



Tip: An IdP adapter instance is available for use within an SP connection only after it has been deployed and configured in PingFederate.

You can use attributes returned from the authentication source (an adapter or a connection mapping 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 292).

To select an adapter instance:

- ▶ Choose an Adapter Instance from the drop-down list and click **Next** to continue.

(Optional) To override any IdP adapter instance, select the **Override Instance Settings** checkbox (see “[Overriding IdP Adapter Instances](#)” on page 289).

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. However, the same adapter instance may be mapped by multiple connections.

To select a connection mapping contract:

- ▶ Choose a Connection Mapping Contract from the drop-down list and click **Next** to continue.

If the connection mapping you need is not available, click **Manage Connection Mapping Contracts** to define one or more connection mapping contracts you need for this connection.

Note that a connection mapping contract can be mapped only once per connection. You may however map the same contract to multiple connections.

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Adapter Instance** or **Connection Mapping Contract** 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 18 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 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:

Override Screen	Manual Section
Adapter	See “ Configuring an IdP Adapter ” on page 252.
Actions	See “ Invoking Adapter Actions ” on page 253.
Extended Contract	See “ Extending an Adapter Contract ” on page 253.
Adapter Attributes	See “ Setting Pseudonym Values and Masking ” on page 254.

- 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.
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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Adapter Instance** on the Summary screen.
9. Select the **Override Instance Settings** checkbox.
10. Click **Next**.

Restricting an Authentication Source to Certain Virtual Server IDs

When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), you have the option to enforce authentication requirements by restricting an authentication source (an adapter or a connection mapping contract) to certain virtual server IDs. This optional setting can be applied to each authentication source added to the connection using virtual server IDs. By default, no restriction is imposed.

Restrict Virtual Server IDs

To restrict an authentication source 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 virtual server IDs that you want to allow for this authentication source.
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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source 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 38.

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 authentication source (an IdP adapter or a connection mapping contract).

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.



- If you use only the Adapter Contract or Connection Mapping Contract values, then you map values for the attribute contract next (see [“Mapping Default Attribute Contract Fulfillment”](#) on page 311).
- 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 292).

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source 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 or the connection mapping contract (see [“Creating an Attribute Contract”](#) on page 285 and [“Authentication Source Mapping”](#) on page 287). 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 311.)

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 285). Attribute sources can be reused across connections to other SP partners.

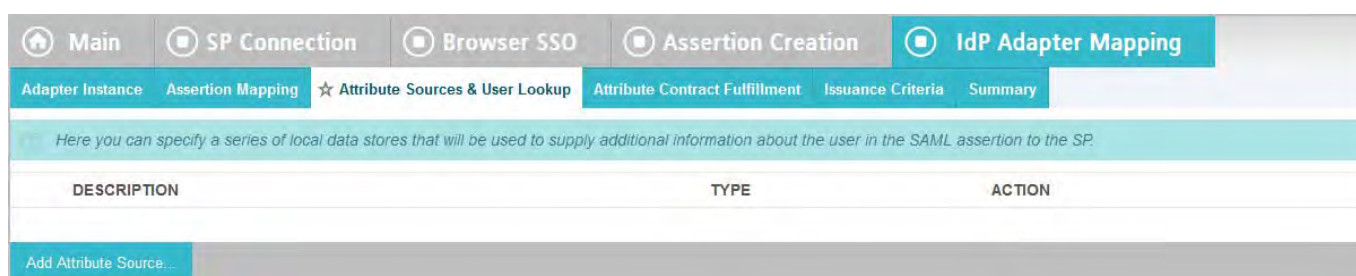
You can use more than one attribute source when mapping values to an assertion:

- 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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Contract Mapping tab.

If this step is not listed, then this instance is configured to use values from the authentication source (an adapter or a connection mapping contract) only (see “[Selecting Assertion Mapping](#)” on page 292).

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:

Data Store Type	Related Manual Section
JDBC	<ul style="list-style-type: none">• “Selecting a JDBC Database Table and Columns” on page 295• “Configuring a Database Filter (WHERE Clause)” on page 297
LDAP	<ul style="list-style-type: none">• “Configuring an LDAP Directory Search” on page 299• “Configuring an LDAP Filter” on page 302
Custom	<ul style="list-style-type: none">• “Configuring Custom Source Filters” on page 304• “Selecting Custom Source Fields” on page 304

3. See [“Attribute Contract Fulfillment”](#) on page 304.

Selecting a Data Store

This screen allows you to choose a data store from a previously configured list (see [“Managing Data Stores”](#) on page 122). Attribute values extracted from this data store are used to help fulfill the attribute contract or the connection mapping contract for this partner (see [“Creating an Attribute Contract”](#) on page 285).

The screenshot shows the 'Data Store' configuration screen in PingFederate. At the top, there is a navigation bar with tabs: 'Main', 'SP Connection', 'Browser SSO', 'Assertion Creation', and 'IdP Adapter Mapping'. Below the navigation bar, the 'Data Store' tab is selected, and the 'Summary' sub-tab is active. A teal banner contains the text: 'This server uses local data stores to retrieve supplemental attributes to be sent in an assertion. Specify an Attribute Source name that will distinguish this user lookup for the selected data store.' Below the banner, there are four input fields: 'Attribute Source Id' (text input), 'Attribute Source Description' (text input), 'Active Data Store' (dropdown menu showing '- SELECT -'), and 'Data Store Type' (text input showing 'None'). At the bottom left, there is a button labeled 'Manage Data Stores...'. The 'Browser SSO' and 'Assertion Creation' tabs are highlighted in grey.

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.

If this step is not shown, you have elected not to look up attributes in data stores (see [“Selecting Assertion Mapping”](#) on page 292).

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 292).

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 304).

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 122).

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](#)

an [Attribute Contract](#)” on page 285). 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:

```
jdbc:mysql://myhost.mydomain.com:3306/
pf?sessionVariables=sql_mode=ANSI_QUOTES
```

For more information, see:

dev.mysql.com/doc/refman/5.0/en/identifiers.html
and
dev.mysql.com/doc/refman/5.1/en/option-files.html

The screenshot shows the 'Database Table and Columns' configuration page in PingFederate. The navigation bar includes 'Main', 'SP Connection', 'Browser SSO', 'Assertion Creation', and 'IdP Adapter Mapping'. The current page has sub-tabs: 'Data Store', 'Database Table and Columns' (selected), 'Database Filter', 'Attribute Contract Fulfillment', and 'Summary'. A message states: 'Please select the table and columns you want to query. This information, along with the attributes supplied in the contract, will be used to fulfill the contract.' The form includes a 'Schema' dropdown set to 'dbo', a 'Table' dropdown set to 'USERS', and a section for 'Columns to return from SELECT' with 'EMAIL' and 'FIRSTNAME' listed. There are 'Remove' and 'Add Attribute' buttons, and a 'Refresh' button. A 'View Attribute Contract' link is at the bottom.

Field Descriptions

Field	Description
Schema	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.
Table	Displays the table(s) contained in the database. Select the table to retrieve data from the data store.
Columns to return from SELECT	Displays the columns available from the selected table. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.
If this step is not shown, you have elected not to look up attributes in data stores (see [“Selecting Assertion Mapping”](#) on page 292).
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 285). Then determine what information is coming in from the session lookup (see [“Selecting Assertion Mapping”](#) on page 292). Information not contained in the authentication source (an adapter contract or a connection mapping 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] [O...]
```


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 295).

The right side generally uses values passed in from your authentication source (an adapter or a connection mapping contract).



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 24.

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.

Main	SP Connection	Browser SSO	Assertion Creation	IdP Adapter Mapping
Data Store	Database Table and Columns	★ Database Filter	Attribute Contract Fulfillment	Summary
Please supply a WHERE clause to filter the data from your table.				
Where				
<input type="text"/>				
Adapter Values				
\${lname}				
\${username}				
\${email}				
\${sessionId}				
\${subject}				
\${fname}				
View List of Columns from "USERS" table				

Field Descriptions

Field	Description
Where	WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: WHERE email='clive@company.com'.

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 or Connection Mapping Contract tab.
If this step is not shown, you have elected not to look up attributes in data stores (see [“Selecting Assertion Mapping”](#) on page 292).
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 304).

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.

The screenshot shows the 'Summary' tab of the 'LDAP Directory Search' configuration. At the top, there are navigation tabs: Main, SP Connection, Browser SSO, Assertion Creation, and IdP Adapter Mapping. Below these are sub-tabs: Data Store, LDAP Directory Search (selected), LDAP Filter, Attribute Contract Fulfillment, and Summary. A teal banner contains the instruction: 'Please configure your directory search. This information, along with the attributes supplied in the contract, will be used to fulfill the contract.'

The configuration fields are:

- Base DN:** An empty text input field.
- Search Scope:** A dropdown menu currently set to 'Subtree'.
- Attributes to return from search:** A table with three columns: 'ROOT OBJECT CLASS', 'ATTRIBUTE', and 'ACTION'.

ROOT OBJECT CLASS	ATTRIBUTE	ACTION
- SELECT -	Subject DN	Add Attribute

At the bottom left, there is a link labeled 'View Attribute Contract'.

Field Descriptions

Field	Description
Base DN	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.
Search Scope	Determines the node depth of the query. Select Subtree, One level or Object.
Root Object Class	The class containing the attributes you want.
Attributes to return from search	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.

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.

8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.

If this step is not shown, you have elected not to look up attributes in data stores (see “[Selecting Assertion Mapping](#)” on page 292).

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 294.

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.



Note: When connecting to Microsoft Active Directory, if you choose the memberOf attribute, an optional checkbox, Nested Groups, appears on the right. Select this checkbox if you want PingFederate to query for groups the end users belong to directly as well as indirectly through nested group membership (if any) under the Base DN.

For example, suppose you have three groups under the Base DN, namely Canada, Washington and Seattle. Seattle is a member of Washington. Ana Smith is an end user and a member of Seattle. If the Nested Groups checkbox is selected, when PingFederate queries for Ana’s memberOf attribute values, the expected results are Seattle and Washington. (When the Nested Groups checkbox is not selected (the default), the expected result is Seattle.)

For Oracle Directory Server, choose isMemberOf under Attribute for nested group membership. For more information, see [documentation about isMemberOf from Oracle](https://docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm) (docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm).

5. Repeat the last step for other attributes as needed.



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.



Tip: If you need to include tokenGroups as one of the attributes, select Object as the search scope and enter a Base DN matching the Subject DN of the authenticated user—You can use variables from the authentication source (an adapter or a connection mapping contract) or results from the previous lookup in the Base DN to fulfill this requirement.

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, Hex or SID) 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.

Claims-based authentication with Microsoft Outlook Web App and Exchange admin center (EAC) is another example. It requires tokenGroups (another binary attribute in Active Directory) to be SID-encoded.



Note: Attributes are specified as binary when configuring an LDAP connection (see “[Specifying LDAP Binary Attributes](#)” on page 132).

ATTRIBUTE NAME	ATTRIBUTE ENCODING TYPE
objectGUID	Base64

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.
If this step is not shown, you have elected not to look up attributes in data stores (see “[Selecting Assertion Mapping](#)” on page 292).
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 299).

The right side generally uses values passed in from your authentication source (an adapter or a connection mapping contract).



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 24.

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

Field	Description
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 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.

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.

If this step is not shown, you have elected not to look up attributes in data stores (see [“Selecting Assertion Mapping”](#) on page 292).

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 294.

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 127).

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 [attribute 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 285). 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.

Main SP Connection Browser SSO Assertion Creation IdP Adapter Mapping			
Adapter Instance Assertion Mapping Attribute Sources & User Lookup ★ Attribute Contract Fulfillment Issuance Criteria Summary			
Fulfill your Attribute Contract with values from one or more data stores, the authentication adapter, or dynamic text values.			
ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS
SAML_SUBJECT	Adapter	subject	None available
email	JDBC (usersdb)	EMAIL	None available
fname	Adapter	fname	None available
lname	Adapter	lname	None available



Note: This is an example screen. The screen presented may look and behave differently depending on how you are mapping assertions (see [“Selecting Assertion Mapping”](#) on page 292).

Map each attribute to fulfill the Attribute Contract from one of these Sources:

- Adapter or Connection Mapping Contract (the authentication source)
 Values are returned from the authentication source. When you make this selection, the associated Value drop-down list is populated by the authentication source (an adapter or a connection mapping contract).
 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.



Important: If you are configuring an SP connection to bridge one or more identity providers to a service provider, consider mapping the original issuer of the assertions into an attribute by selecting Context as the **Source** and Authenticating Authority as the **Value**. This is especially important when bridging multiple identity providers to one service provider—the service provider should take the information about the original issuer into consideration before granting access to protected resources.

See [“Federation Hub”](#) on page 42 and [“Bridging multiple IdPs to an SP”](#) on page 44 for more information.



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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- LDAP/JDBC/Custom



Note: PingFederate appends a description in parentheses for data stores of the same type (see [“Selecting a Data Store”](#) on page 294).

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 299, [“Selecting a JDBC Database Table and Columns”](#) on page 295, or [“Configuring Custom Source Filters”](#) on page 304).

- 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 613). 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 authentication source (an adapter or a connection mapping contract), 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 294) 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 563).

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
8. Click **Attribute Sources & User Lookup** under the IdP Adapter Mapping or Connection Mapping Contract tab.
If this step is not shown, you have elected not to look up attributes in data stores (see [“Selecting Assertion Mapping”](#) on page 292).
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 294).

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 25). This token authorization can be used to restrict who can SSO to protected resources.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT - *	- SELECT - *	- SELECT - *	*		Add



Note: This screen does not appear if you chose to use data stores with the failsafe mapping option (see [“Selecting Assertion Mapping”](#) on page 292).

To configure Issuance Criteria:

1. Select a source containing attributes for token authorization.
Associated attributes appear in the Attribute Name drop-down list:

- Adapter or Connection Mapping Contract– Select to access attributes from the authentication source.
- 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 (see [“Selecting a Data Store”](#) on page 294).

- Mapped Attributes – Select to access the required attributes.
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 93).

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 93).



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 613).



Important: When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), 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 616).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618. 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 the rest of the 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. (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 292).



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.

Adapter Instance Assertion Mapping Attribute Sources & User Lookup ★ Failsafe Attribute Source Attribute Contract Fulfillment Summary

If attribute mappings configured with the data-store setup fail to complete the Attribute Contract, PingFederate can either provide a default set of attributes (configured on the next screen) or stop the SSO transaction.

Send user to SP using default list of attributes

Abort the SSO transaction

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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source 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 292).

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 304) or if you choose not to set up attribute sources (see [“Selecting Assertion Mapping”](#) on page 292).

Map each attribute to fulfill the Attribute Contract from one of these Sources:

- Adapter or Connection Mapping Contract (the authentication source)
Values are returned from the authentication source. When you make this selection, the associated Value drop-down list is populated by the authentication source (an adapter or a connection mapping contract).
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.



Important: If you are configuring an SP connection to bridge one or more identity providers to a service provider, consider mapping the original issuer of the assertions into an attribute by selecting Context as the **Source** and Authenticating Authority as the **Value**. This is especially important when bridging multiple identity providers to one service provider—the service provider should take the information about the original issuer into consideration before granting access to protected resources.

See [“Federation Hub”](#) on page 42 and [“Bridging multiple IdPs to an SP”](#) on page 44 for more information.



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 618). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 613. For syntax and examples, see sections under “Constructing Expressions” on page 614.)

- 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 613). 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 authentication source (an adapter or a connection mapping contract), using the `${attribute}` syntax.



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 563).

There are a variety of reasons that you might not have a text value. For example, if your OIDC Web application provides a consumer service, you might want to supply a specific parameter value.

Main		SP Connection		Browser SSO		Assertion Creation		IdP Adapter Mapping	
Adapter Instance		Assertion Mapping		Attribute Sources & User Lookup		Failsafe Attribute Source		★ Attribute Contract Fulfillment	
								Summary	
<p>Fulfill your Attribute Contract with values from the authentication adapter or dynamic text values. These values will be used if the user cannot be located in data stores. Coordinate with your partner to determine what default values to send.</p>									
ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS						
SAML_SUBJECT	Adapter	subject	None available						
email	Adapter	email	None available						
fname	Adapter	fname	None available						
lname	Adapter	lname	None available						

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.
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 **Authentication Source Mapping** on the Summary screen.
7. Click the authentication source name.
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 310.

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 280). The screen also displays configured information.

(For WS-Federation, the configuration of bindings is not applicable.)

This task provides the configuration for specific endpoints and security considerations applicable to selected profiles. Click the button below to create or revise this configuration.

Protocol Settings	
Outbound SSO Bindings	POST, Artifact
Outbound SLO Bindings	POST, Redirect, Artifact, SOAP
Inbound Bindings	POST, Redirect, Artifact, SOAP
Artifact Lifetime	60 second(s)
Signature Policy	SAML-standard, Authn requests over POST & Redirect
Encryption Policy	No Encryption

Configure Protocol Settings

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 280).



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.

Home Main
SP Connection
Browser SSO
Protocol Settings

★ Assertion Consumer Service URL
SLO Service URLs
Allowable SAML Bindings
Artifact Lifetime
Artifact Resolver Locations
Signature Policy
Encryption Policy

Summary

As the IdP, you send SAML assertions to the SP's **Assertion Consumer Service**. The SP may request that the SAML assertion be sent to one of several URLs, via different bindings. Please provide the possible assertion consumer URLs below and select one to be the default.

DEFAULT	INDEX	BINDING	ENDPOINT URL	ACTION
	0	Artifact	/sp/ACS.saml2	Edit / Delete
default	1	POST	/sp/ACS.saml2	Edit / Delete
<input type="checkbox"/>	<input type="text"/>	- SELECT - *	<input type="text"/>	<input type="button" value="Add"/>

Field Descriptions

Field	Description
Default (SAML 2.0)	A check in this checkbox indicates that the URL configuration in that row will be used as a default.
Index (SAML 2.0)	Uniquely identifies multiple ACS endpoints.
Binding	The method of transmission: POST or Artifact.
Endpoint URL	A location to which the assertion is sent, according to partner requirements.

To reach this screen for editing:

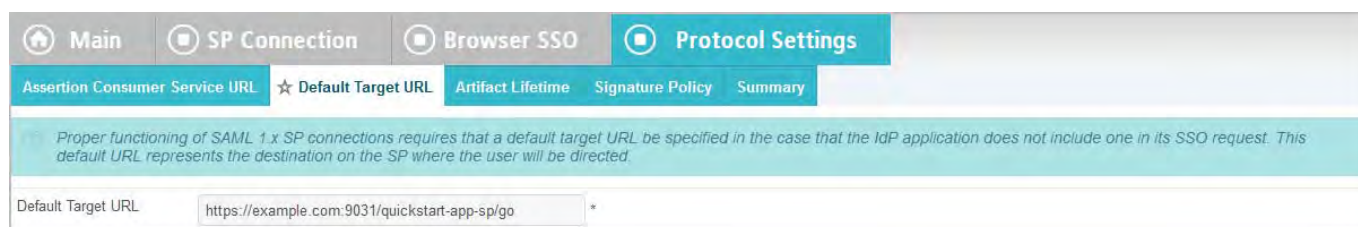
1. Click a connection name on the Main Menu.
Click **Manage All SP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the SP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **Protocol Settings** on the Summary screen.
5. Click **Configure Protocol Settings**.
6. Click **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 276).
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

Field	Description
Default Target URL	The URL of the target SP resource.

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 **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.

To protect against session token hijacking, you can specify additional allowed domains and paths in this screen. If the option to validate wreply for SLO is enabled, these additional domains and paths will also be taken into consideration as well (see [“Managing Partner Redirect Validation”](#) on page 145).

The screenshot shows the 'Protocol Settings' tab in the configuration interface. Under the 'Service URL' section, there is a 'Summary' sub-section with a teal background containing the instruction: 'As the IdP, you send SAML assertions and SLO cleanup messages to the SP. Specify here the URL where the SP is expecting to receive these messages.' Below this is an 'Endpoint URL' field with the value '/sp/prp.wsf'. A note states: 'You can specify additional allowed domains and paths below. This whitelist will also be used to validate wreply for SLO, if enabled under Server Settings > Redirect Validation.' Below the note is a table with columns: 'Require HTTPS', 'Valid Domain Name (leading wildcard '*' allowed)', 'Valid Path (leave blank to allow any path)', 'Allow Any Query/Fragment', and 'Action'. The first row has a checked 'Require HTTPS' checkbox, an empty domain name field, an empty path field, an unchecked 'Allow Any Query/Fragment' checkbox, and an 'Add' button.

To define a service URL:

- ▶ Enter the fully qualified URL or just the relative path if you have defined a base URL (see [“General Information”](#) on page 276). You must include the initial slash if you are entering only a relative path.

To specify additional allowed domains and paths:

1. Indicate whether to require HTTPS.



Important: This selection is recommended to ensure that the validation will always prevent message interception for this type of potential attack, under all conceivable permutations.

2. Enter the expected domain or IP address under Valid Domain Name.

Use the domain only, without qualifiers. For example:

company.com

Using an initial wildcard and period for a domain name will cover multiple subdomains. For example:

*.company.com

covers hr.company.com or email.company.com.

(Optional) Enter the exact path of the resource (case-sensitive) under Valid Path. 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:

/app/Consumer.jsp

allows /app/Consumer.jsp but rejects /app/consumer.jsp.



Tip: You can also enter multiple query parameters with or without a fragment.

For example:

/app/Consumer.jsp?area=West&team=IT#ref1001

matches /app/Consumer.jsp?area=West&team=IT#ref1001 but not /app/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 resource.



Note: When selected, no query parameter and/or fragment is allowed in the path.

Click **Add**.

3. Repeat the previous step to add additional entries as needed.
4. Click **Save**.

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 281 or “Configuring SP-Initiated SLO” on page 282).

BINDING	ENDPOINT URL	RESPONSE URL	ACTION
POST	/sp/SLO.saml2		Edit / Delete
- SELECT -			Add

Field Descriptions

Field	Description
Binding	The method of transmission: POST, Artifact, Redirect, or SOAP.
Endpoint URL	A location to which SLO logout request messages are sent, according to SP requirements.

Field	Description
Response URL	(Optional) A location to which SLO logout response messages are sent according to SP requirements. When omitted, the PingFederate IdP server sends logout responses to the Endpoint URL.

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 **SLO Service URLs** on the Summary screen.

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 276).
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.

The screenshot shows the 'Protocol Settings' tab selected in the navigation menu. Below it, the 'Allowable SAML Bindings' sub-tab is active. A question is posed: 'When the SP sends messages, what SAML bindings do you want to allow?'. Below this question, there are four checkboxes: 'Artifact' (unchecked), 'POST' (checked), 'Redirect' (unchecked), and 'SOAP' (unchecked).

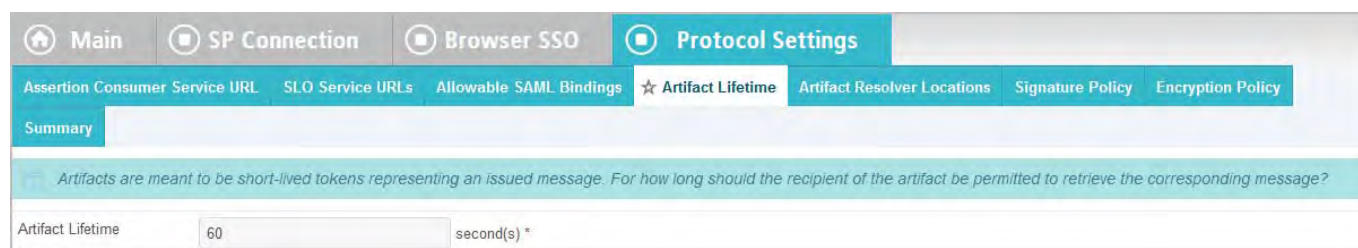
To reach this screen for editing:

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.

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.

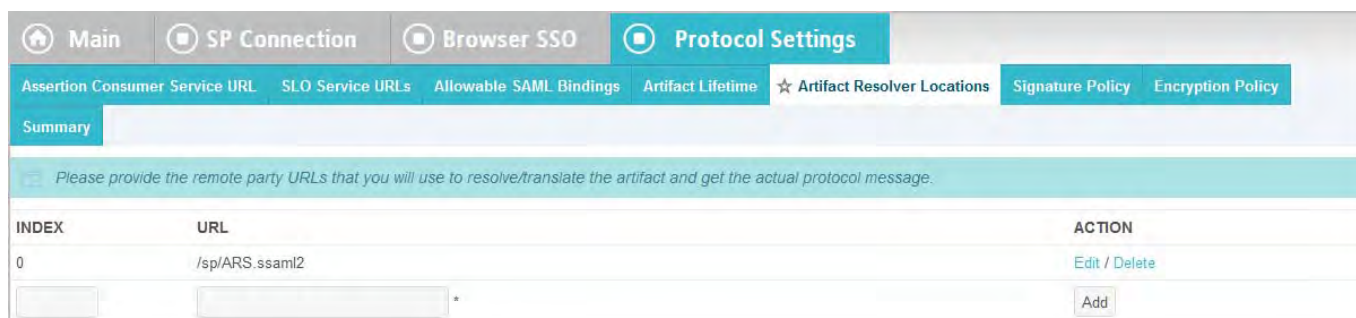
To reach this screen for editing:

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 **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.
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 **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 276).
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 29).

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.)

Home	Main	SP Connection	Browser SSO	Protocol Settings	
Assertion Consumer Service URL	SLO Service URLs	Allowable SAML Bindings	Artifact Lifetime	Artifact Resolver Locations	★ Signature Policy
Encryption Policy					
Summary					
<p>Additional guarantees of authenticity may be agreed upon between you and your partner. For SP-initiated SSO, you can choose to require signed authentication requests sent via the POST or redirect bindings. You can also choose to sign assertions sent to this partner, regardless of the binding used.</p>					
<input type="checkbox"/> Require AuthN requests to be signed when received via the POST or Redirect bindings					
<input type="checkbox"/> Always sign the SAML Assertion					

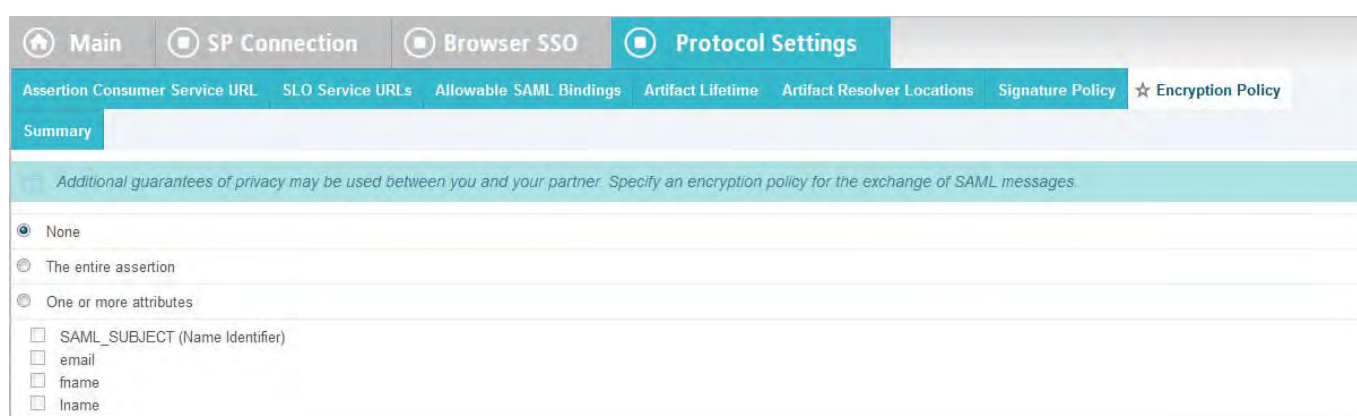
- ▶ Make your selection(s) and click **Next**, or just click **Next** if no additional security is required.

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**.
6. Click **Encryption Policy** on the Summary screen.

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 353).

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.



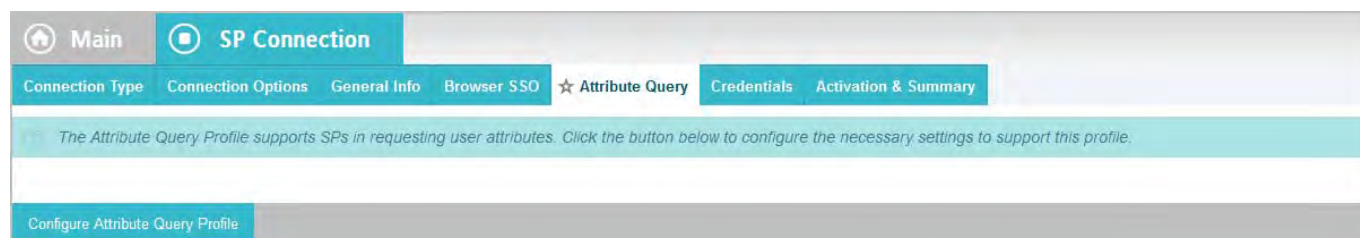
Important: When you finish editing existing settings, be sure to click **Done** on the Summary screen and then **Save** on the Browser SSO screen. For a new connection, click **Done** and then click **Next** on the Browser SSO screen. Save the entire connection on the Activation & Summary screen (see [“Editing and Activating a Connection”](#) on page 353).

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

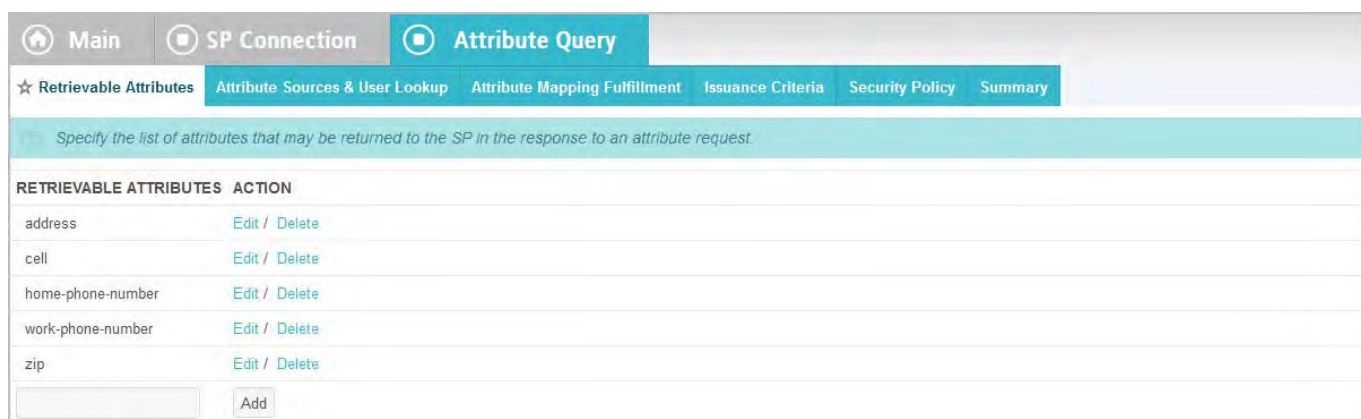
At the Attribute Query step you configure your connection to respond to requests for user attributes from your partner SP, if you have chosen this option (see [“Choosing Connection Options”](#) on page 275). Attribute queries are not dependent on single sign-on but may be used independently or in conjunction 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**.

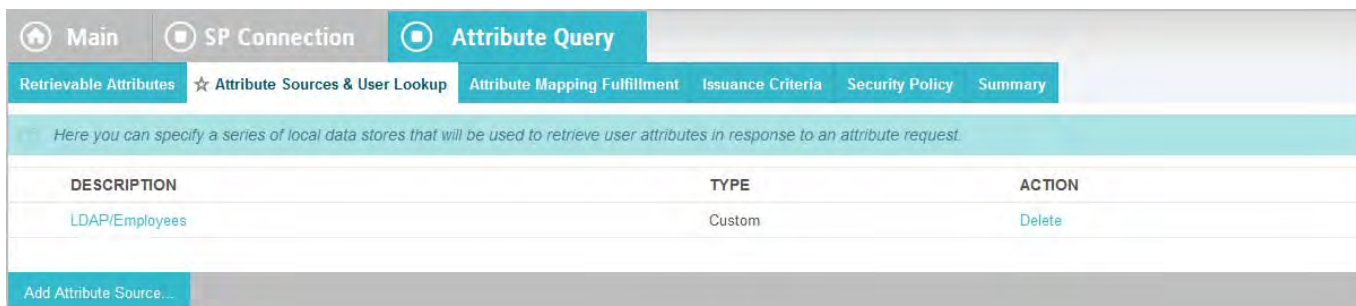
To delete an attribute:

- ▶ 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 324 next).

To modify an attribute source configuration:

1. Click the attribute source Description link.
2. Click **Save** on the screen you change.



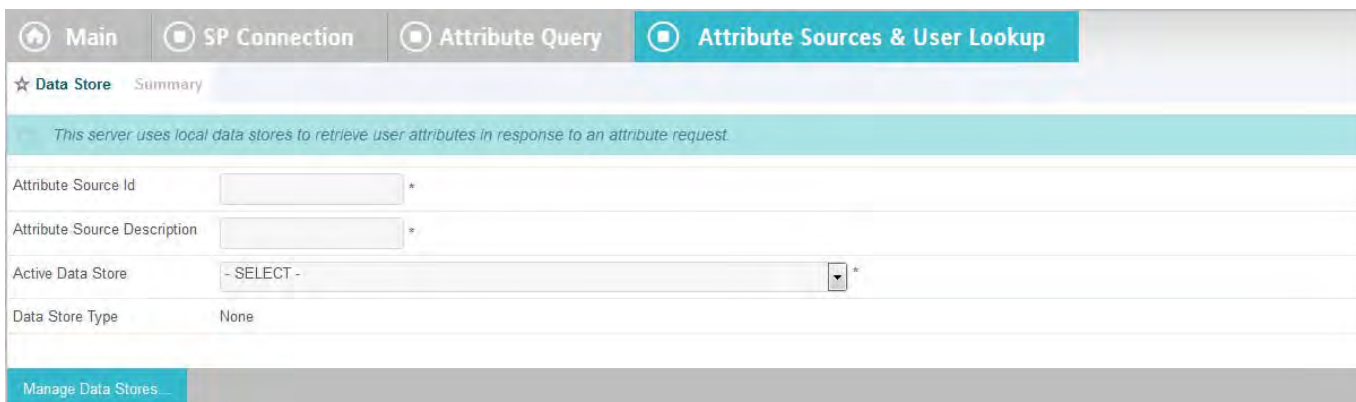
Note: Depending on what you change, you may need to modify dependent data in subsequent steps, as indicated. Click **Save** or **Done** when either of those options appears.

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 “[Authentication Source Mapping](#)” on page 287). Therefore, you must identify the data stores that contain the attributes on your system.



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 122).

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 295
- [“Configuring a Database Filter \(WHERE Clause\)”](#) on page 297

If you use a LDAP data store, see:

- [“Configuring an LDAP Directory Search”](#) on page 299
- [“Configuring an LDAP Filter”](#) on page 302

If you use a Custom data store, see:

- [“Configuring Custom Source Filters”](#) on page 304
- [“Selecting Custom Source Fields”](#) on page 304

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.

Main SP Connection Attribute Query			
Retrievable Attributes Attribute Sources & User Lookup ★ Attribute Mapping Fulfillment Issuance Criteria Security Policy Summary			
Fulfill your Attribute Request with values from your Data Store lookup or with dynamic text values.			
ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS
cell	LDAP (LDAP/Employees)	mobile	None available
email	LDAP (LDAP/Employees)	mail	None available

Map each attribute into the assertion from one of these Sources:

- Context

Values are returned from the context of the transaction at runtime.



Important: If you are configuring an SP connection to bridge one or more identity providers to a service provider, consider mapping the original issuer of the assertions into an attribute by selecting Context as the **Source** and Authenticating Authority as the **Value**. This is especially important when bridging multiple identity providers to one service provider—the service provider should take the information about the original issuer into consideration before granting access to protected resources.

See [“Federation Hub”](#) on page 42 and [“Bridging multiple IdPs to an SP”](#) on page 44 for more information.



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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 324).

- 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 613). 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 324) 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 25). This extension of token authorization can be used to restrict who can access protected resources.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

Show Advanced Criteria

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.



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 613). For syntax and examples, see sections under “[Constructing Expressions](#)” on page 614.)

- 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 each data store lookup (see “[Choosing a User-Data Store](#)” on page 324).

- Mapped Attributes – Select to access retrievable attributes defined for the attribute query profile.

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.

- (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 613).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618.

- 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 330).

To configure attribute-query security policy for this partner:

- Check or clear the checkboxes and click **Next** or **Done**.

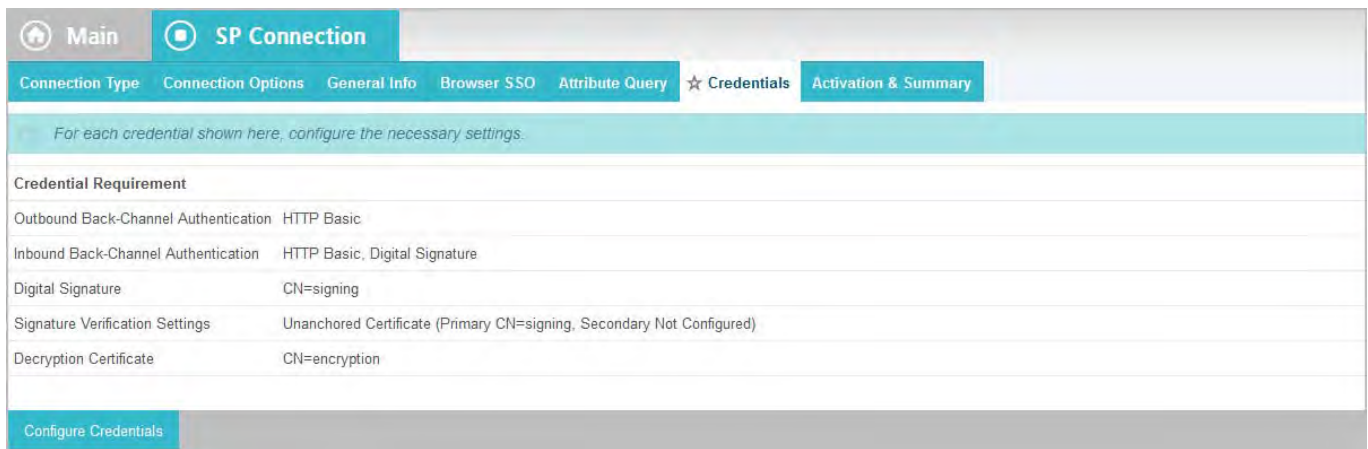
Editing and Saving Attribute Query Configurations

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.



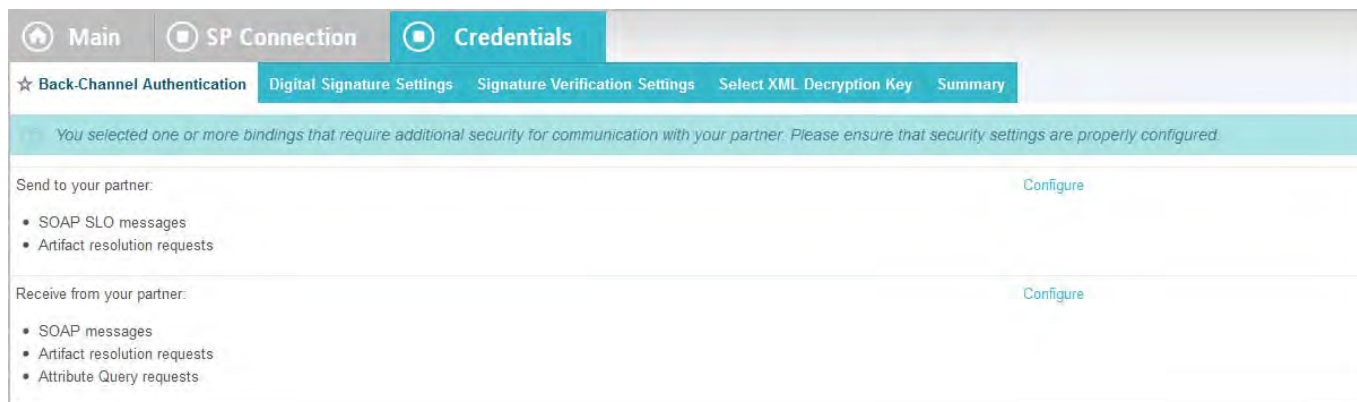
- To continue, click **Configure Credentials**.

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**.
4. Click **Back-Channel Authentication** on the Summary screen.

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 225). You need to export the certificate (only) and send it 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. Optionally, you may also restrict the issuer to a specific Trusted CA to mitigate potential man-in-the-middle attacks as well as to provide a means to isolate certificates used by different connections. The CA’s certificate must be imported into the PingFederate Trusted CA store (see “[Trusted Certificate Authorities](#)” on page 222).
 - 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**.

- 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.

- 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**.

- Click **Next**.
- 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 280) 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 330).



Note: This step does not appear if a connection configuration does not require it.

To reach this screen for editing:

- Click a connection name on the Main Menu.
Click **Manage All SP**, if needed, to see a full list of connections.
- Click **Credentials** under the SP Connection tab.
- Click **Configure Credentials**.

- Click **Digital Signature Settings** on the Summary screen.

To specify a certificate:

- 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 227).
- (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.

- (Optional) Select the Signing Algorithm from the drop-down list.
The default selection is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection if you and your connection partner have agreed to use a stronger algorithm.

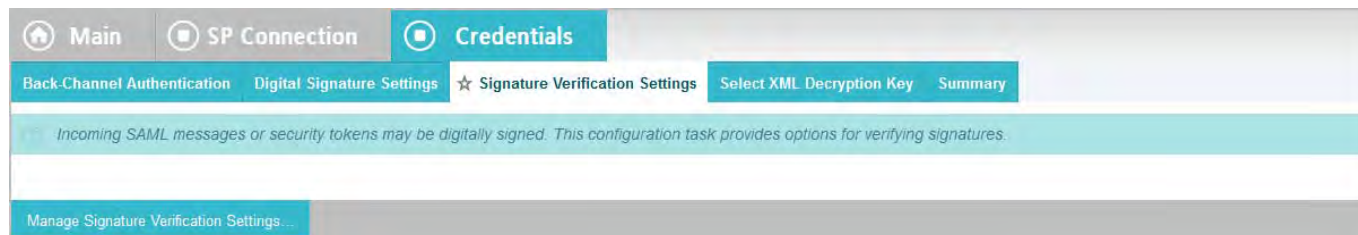
Configuring Signature Verification Settings

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 29). 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 222).

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:

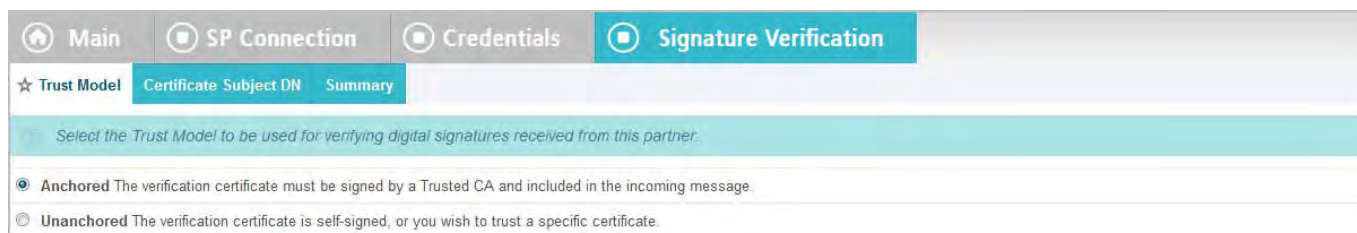
- Click a connection name on the Main Menu.
Click **Manage All SP**, if needed, to see a full list of connections.

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 29).



- ▶ 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, [“Anchored Certificates--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 Unanchored Certificates”](#) on page 336).

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 **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 29). PingFederate verifies that the Issuer DN (if specified) matches that of one of the issuers in the chain, the Issuer CA is trusted and the embedded certificate’s Subject DN matches the one specified on this screen. If so, PingFederate uses that certificate to verify the message signature.

To complete the configuration:

1. Enter the Subject DN or extract it from your SP partner's certificate if the certificate is stored on an accessible file system.



Tip: To extract the Subject DN from a certificate, browse to select your SP partner's public certificate and click **Extract**.

2. (Optional) Select the Restrict Issuer checkbox. Enter the Issuer DN or extract it from a certificate if it is stored on an accessible file system.



Important: We recommend enabling this option to mitigate potential man-in-the-middle attacks as well as to provide a means to isolate certificates used by different connections.

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 **Certificate Subject DN** on the Summary screen.

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 29).

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 321).

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 329).

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**.

If this step is not present, you have chosen not to encrypt the assertion or the SAML_SUBJECT (see “[Configuring XML Encryption Policy \(SAML 2.0\)](#)” on page 321).

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](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (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/)” (<http://www.w3.org/TR/xmlenc-core/>).



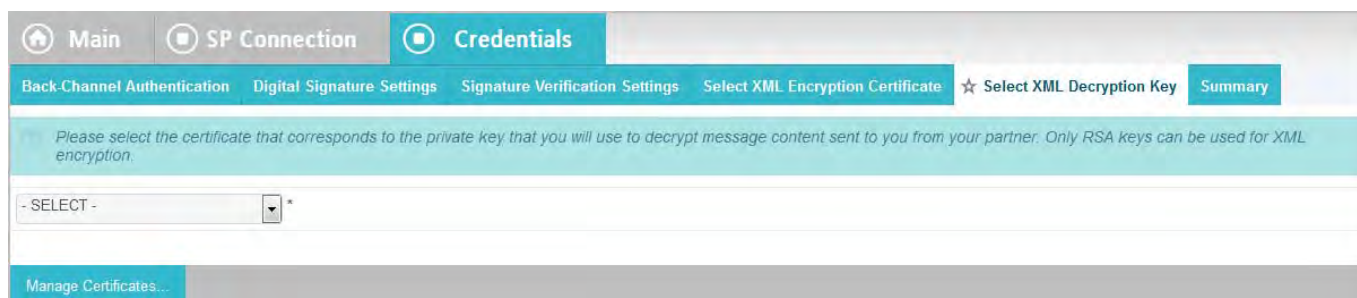
Note: As a Key Transport Algorithm, RSA-v1.5 is disabled for new connections for security reasons. If you are updating an existing connection that uses RSA-v1.5, we recommend changing the selection for increased security.

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 321).

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 227).

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 35).



Note: This configuration task is presented in the administrative console only when you enable Outbound Provisioning (see [“Choosing a Connection Type”](#) on page 273).

The screenshot shows the 'SP Connection' configuration page with the 'Outbound Provisioning' tab selected. The page has a breadcrumb trail: Main > SP Connection > Connection Template > Connection Type > General Info > Outbound Provisioning > Activation & Summary. A teal banner at the top reads 'Configure outbound provisioning for this service provider.' Below this, the 'Outbound Provisioning' section contains two rows: 'Connection Type' with the value 'SCIM 1.1 Service Provider' and 'Source Repository' with the value 'Not Configured'. At the bottom left, there is a 'Configure Provisioning' button.

► 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.

The screenshot shows the 'Configure Channels' page with the 'Target' tab selected. A teal banner at the top reads 'Specify credentials and/or other connection details that PingFederate will use to access the target service provider for outbound provisioning.' Below this, the 'Provisioning Target' section contains several fields: 'Users Resource URL' (with an asterisk), 'Groups Resource URL', 'Authentication Method' (with radio buttons for None, Basic Authentication, and OAuth 2.0 Bearer Token, where OAuth 2.0 Bearer Token is selected), 'User', 'Password', 'Client ID', 'Client Secret', and 'Token Endpoint URL'. At the bottom, there are two checkboxes: 'SCIM SP supports PATCH updates' and 'Provision groups with distinguished name', both of which are unchecked. The 'Deprovision Method' section has radio buttons for 'Delete User' and 'Disable User', with 'Disable User' selected.



Note: The target configuration settings represented in this screen example and described in the procedure below are for the bundled PingFederate provisioning plug-in for SCIM providers. For SaaS Connector targets, please refer to documentation in your add-on distribution package, or [locate user guides online](#) (documentation.pingidentity.com). For SCIM provisioning to [PingOne](#), an administrator can find target information by setting up an Identity Repository and enabling provisioning, or by inspecting an existing setup.

To configure the Target:



Note: First obtain the Users Resource URL from your service provider—for example, <https://example.com/v1/Users>. If supported by the provider, also obtain the Groups Resource URL—for example, <https://example.com/v1/Groups>. Also determine whether the provider supports PATCH updates (see [Step 5](#) below) and/or prefers full distinguished name (DN)s for group provisioning (see [Step 6](#)).

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 DN)s, rather than only Common Names (CN)s, to identify groups.
Some SCIM providers, including [PingOne](#) (Ping Identity's cloud identity management service), allow administrators to parse full DN)s when necessary—for example, in the case of duplicate CN)s—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**.



Note: For some provisioning plug-ins, when you first enter or change credentials and click **Next**, PingFederate immediately tests connectivity to the target.

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 340). 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 111).
2. On the Outbound Provisioning screen, click **Configure Provisioning**.

To configure a new channel:

- ▶ 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 343).
3. Make any further changes needed for the new channel.

To edit a channel:

- ▶ Click the Channel Name link.

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 352).

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.



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 127).

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 348).

- ▶ 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 127).

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.

Main		SP Connection		Configure Channels		Channel	
Channel Info		Source		★ Source Settings		Source Location	
		Attribute Mapping		Activation & Summary			
<input type="checkbox"/> Enter or modify LDAP settings that apply to the source user-data store, as needed. Note that these fields are preconfigured with default settings based on the LDAP Type, when specified (see documentation). For most LDAP Directory installations, the default settings can be used.							
Data Source							
Data Source	tpdc1.techpub.local						
Data Source Description	Active Directory						
Ldap Type	ActiveDirectory						
Identity							
Entry GUID Attribute	objectGUID *						
GUID Type	Binary ▾						
Group Membership Detection							
Member of Group Attribute	memberof *						
Group Member Attribute	member *						
Change Detection							
User objectClass	user *						
Group objectClass	group *						
Changed Users/Groups Algorithm	Active Directory USN ▾						
USN Attribute	uSNChanged *						
Timestamp Attribute	modifyTimestamp *						
Account Management							
Account Status Attribute	userAccountControl *						
Account Status Algorithm	Active Directory Bitmap ▾						
Default Status	true ▾						
Flag Comparison Value	<input type="text"/>						
Flag Comparison Status	false ▾						

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 129).

Field Descriptions

Field	Description
Entry GUID Attribute	The name of the attribute in the data store representing the user's Globally Unique Identifier.
GUID Type	Indicates whether the GUID is stored in binary or text format. Active Directory is always binary. Other LDAP stores most often use text.
Member of Group Attribute	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.
Group Member Attribute	The name of a multi-value group attribute used to track membership in the group using either DN or GUID values.
User objectClass	The LDAP object class to which user entries belong, used to restrict search results to user entries only.
Group objectClass	The LDAP object class to which group entries belong, used to restrict search results to group entries only.
Changed Users/Groups Algorithm	<p>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:</p> <p>Active Directory USN – For Active Directory only, this algorithm queries for update sequence numbers on user records that are larger than the last time records were checked.</p> <p>Timestamp – Queries for timestamps on user records that are <i>not older</i> 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.</p> <p>Timestamp No Negation – Queries for timestamps on user records that are <i>newer</i> than the last time records were checked. This algorithm is recommended for the Oracle Directory Server.</p>
USN Attribute	The name of the attribute used to store the update sequence number—applicable when the Active Directory algorithm is chosen above.
Timestamp Attribute	The name of the attribute used to store the timestamp on user records.

Field	Description
Account Status Attribute	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> .
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 <code>userAccountControl</code> flags, see Microsoft's knowledge base (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, <code>nsaccountlock</code>) 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 <i>true</i> equals enabled while setting the value to <i>false</i> equals disabled. For example, if the Account Status Attribute is set to <code>nsaccountlock</code> , and the Flag Comparison Value is set to true , and the Flag Comparison Status is set to false , then any users with <code>nsaccountlock=true</code> 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's for maintaining corresponding groups at the service provider.



Note: Groups 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](https://pingidentity.com/support-and-downloads/support-request-form.cfm) (pingidentity.com/support-and-downloads/support-request-form.cfm) for more information.

After specifying the required Base DN, you have the options to provision users and groups (when applicable) based on group membership information or LDAP search results.

Field Descriptions

Field	Description
Base DN	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.

Field	Description
Group DN	<p>For Users, the group Distinguished Name associated with the user store, if applicable—required if a Filter is not used (see “Filter” below).</p> <p>For Groups (optional and subject to support from your partner or the SaaS Connectors), the DN of the higher-level group that contains the user groups, if applicable.</p> <p>Important: If a Group DN is used for Active Directory Groups, the domain functional level must be set as follows:</p> <ul style="list-style-type: none"> • For Windows Server 2003 – Windows 2000 native or Windows Server 2003. • For Windows Server 2008 – Either of the above or Windows Server 2008. • For Windows Server 2012 – Any of the above or Windows Server 2012. <p>Refer to Windows Server documentation or support for more information.</p> <p>(Optional) Select the Nested Search checkbox to include users (and groups) that are members of nested groups of the specified group.</p> <p>Note: Nested Search is available when Active Directory or Oracle Directory Server is selected as the source user repository (see “Identifying the Source Data Store” on page 343).</p>
Filter	<p>An LDAP search filter. For Users, required if a Group DN is not provided.</p> <p>When configuring groups provisioning, an LDAP search filter is required if a group DN (for Groups) is not provided. If both Group DN and Filter are empty, no group is provisioned.</p> <p>For information about LDAP filters, refer to your LDAP documentation. Note that you may need to escape any special characters.</p> <p>Important: Group DN is ignored when an LDAP search filter is provided.</p>

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.

Channel			
Channel Info			
Source			
Source Settings			
Source Location			
★ Attribute Mapping			
Activation & Summary			
<p><i>Edit the mapping of attributes from the local data store into Fields specified by the service provider. The Refresh Fields button queries the target partner to update Fields and specifications.</i></p>			
FIELD NAME	ATTRIBUTE(S)	DEFAULT	ACTION
Username*	- sAMAccountName		Edit
Formatted Name			Edit
Family Name	- sn		Edit
Given Name	- givenName		Edit
Middle Name	- middleName		Edit
Honorific Prefix			Edit
Honorific Suffix	- generationQualifier		Edit
Display Name	- displayName		Edit
Nickname			Edit
Profile URL			Edit
Title	- title		Edit
User Type			Edit
Preferred Language			Edit
Locale			Edit
Timezone			Edit
Password			Edit

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.

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.



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 613).

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).



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.

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.
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 613.



Important: The use of OGNL expressions may not be enabled for your PingFederate installation (see [“Enabling and Disabling Expressions”](#) on page 613).

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).

Mask Log Values – Determines whether sensitive information (for example, the Password attribute) will be masked in PingFederate log files, or not.

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 353).



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 119). 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 563.

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 19).

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 “[Authentication Source Mapping](#)” on page 287).

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:

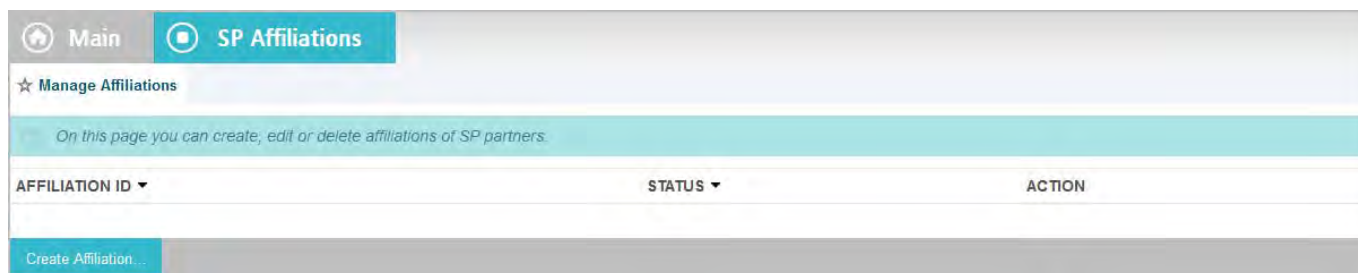
1. Click **Manage All Affiliations** under SP Affiliations on the Main Menu.
2. Click **Delete** under Action for the affiliation you want to delete.
3. Click **Save** to confirm the deletion (or click **undelete**).

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.

Field Descriptions

Field	Description
Affiliation ID	A unique identifier for this affiliation. This value serves as the Name ID qualifier for SAML assertions sent to affiliated SP partners.
Affiliation Owner	Any SAML 2.0 SP connection may serve as the Owner.

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.

SP CONNECTION NAME	ACTION
- SELECT -	Add

- ▶ 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 [“Authentication Source Mapping”](#) on page 287).

- ▶ 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.

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 32). 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 111).

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.

The screenshot shows the 'SP Auto-Connect' configuration page. The 'Assertion Lifetime' tab is selected, showing two input fields: 'Minutes Before' and 'Minutes After', both set to 5. A note above the fields states: 'When an assertion is issued to the SP, there is a timeframe (before and after issuance) of validity. Please specify these parameters below.'

Field Descriptions

Field	Description
Minutes Before	The amount of time before the assertion was issued during which it is to be considered valid.
Minutes After	The amount of time after the assertion was issued during which it is to be considered valid.

To change the default times:

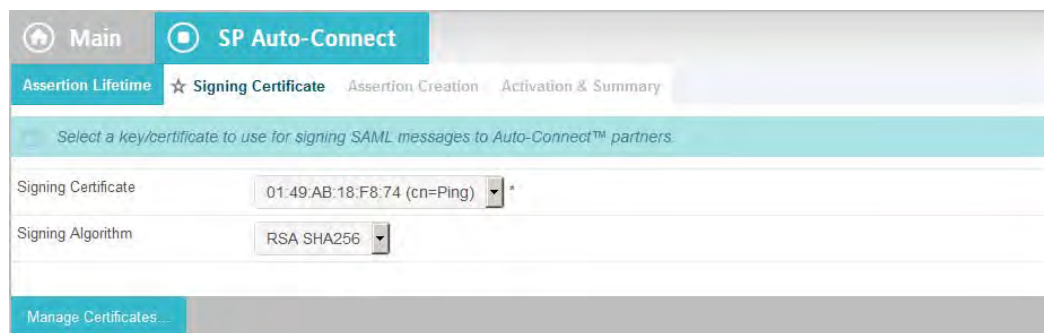
- ▶ (Optional) Edit the desired setting(s) and click **Next** or **Save**.

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 32); there is no need to exchange certificates with your partners.



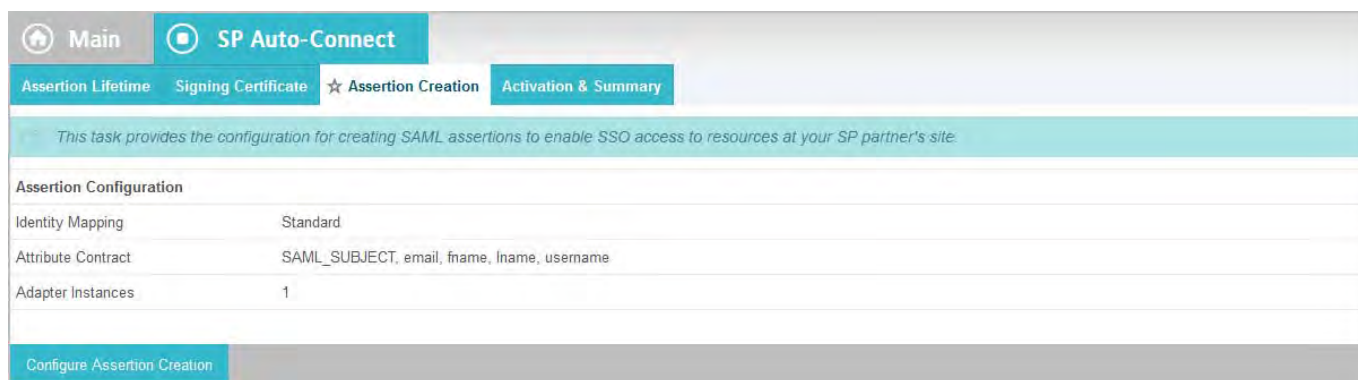
You can use the same certificate used for signing metadata (see “[Configuring Auto-Connect Metadata Signing](#)” on page 120). If you use a different certificate, ensure that it meets Auto-Connect validation requirements (see “[Auto-Connect Security Model](#)” on page 34).

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 227).
2. (Optional) Select the Signing Algorithm from the drop-down list.
The default selection is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection 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 283.

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 121).

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.

DOMAIN NAME	ACTION
pingidentity.com	Edit / Delete
<input type="text"/>	<input type="button" value="Add"/>

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 32.) Alternatively, if an Auto-Connect partner elects not to use the standard location, you can supply the applicable URL.

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 111).

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 38).



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 469.

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 32).

You can also deploy an IdP connection to bridge an identity provider to one or more service providers through one or more connection mapping contracts (see [“Federation Hub”](#) on page 42 and [“Connection Mapping Contracts”](#) on page 162 for more information).

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 15).

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 509).

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 15). 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 534.

If you configure more than one adapter instance, then you must map a target URL to at least one instance (see [“Configuring Target URL Mapping”](#) on page 366).

SP adapter setup is available only if your server is configured as an SP (see [“Choosing Roles and Protocols”](#) on page 111).



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 377). 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

Field	Description
Instance Name	A descriptive name for the adapter instance—for example, an identity management system name.
Instance ID	An internal identifier for the adapter instance. Must be alphanumeric with no spaces.
Type	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.
Parent Instance (Optional)	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.

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 15).

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 18). 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 15). For information about configuring the adapter packaged with PingFederate, see [“Configuring the SP OpenToken Adapter”](#) on page 538.

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 15).

- ▶ For information about configuring the OpenToken Adapter distributed with PingFederate, see “[Configuring the SP OpenToken Adapter](#)” on page 538.
- ▶ 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 533.

ACTION NAME	ACTION DESCRIPTION	ACTION INVOCATION LINK
Download	Download the configuration file for the agent.	Invoke Download

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.

CORE CONTRACT
 subject

EXTEND THE CONTRACT ACTION

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 [“Configuring Target URL Mapping”](#) next).

Configuring Target URL Mapping

When you have more than one target session defined in an IdP connection, you must map the target URL to its target session. During SSO, the target URL requested will be compared against the URL(s) listed here until a match is found. If a match is not found, SSO will fail. Use a wildcard (*) to match multiple URLs to the same target session.

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 567). (When this information is provided with the SP request, the mapping table is ignored.)

Furthermore, when bridging an identity provider to multiple service providers, for each service provider supporting the SAML IdP-initiated SSO profile, map the target URLs to the corresponding SP connection.



Tip: In this scenario, PingFederate is a federation hub for the identity provider and the service providers. (See [“Federation Hub”](#) on page 42 and [“Bridging multiple IdPs to an SP”](#) on page 44 for more information.)

Finally, if an IdP connection is associated with one (or more) SP adapter(s) and one (or more) connection mapping contract(s), you also need to map the target URLs to their respective target session.

Field Descriptions

Field	Description
URL	The target URLs that align with your configured target sessions. The URLs instruct the PingFederate SP server to route session-creation processing through an SP adapter instance or an SP connection. 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. Note: If a target session is not allowed based on restrictions imposed (see "Restricting a Target Session to Certain Virtual Server IDs" on page 396), PingFederate tries the next entry.
Target Type	The type of the Target Session. If the IdP role is not activated (or is activated without any protocol for browser SSO, such as SAML or WS-Federation), Target Type defaults to "SP Adapter".
Target Session	A selection of configured SP adapter instances or SP connections. The available values depends on the chosen Target Type.

The order of mapping is significant in that the first matching mapping, from top to bottom, determines which target session receives the request. For example, if two URLs are mapped in the following order:

URL	Session Target
http://sp.company.com/acct101/	SP OpenToken
http://sp.company.com/*	Fedhub Cxn to SP

A target URL of `http://sp.company.com/acct101/start` will be mapped to 'SP OpenToken' because the target matches the first mapping in the configuration.

If the order of the mappings is reversed, the same target will be mapped to 'Fedhub Cxn to SP' because the first mapping in the new configuration (`http://sp.company.com/*`) matches the target URL

Note that you can use only one wildcard (*) per URL.

To reach this screen for editing:

- ▶ On the Main Menu under Application Integration Settings for SP Configuration, click **Target URL Mapping**.

To create target URL mappings:

1. Enter a URL.
2. Select a Target Type. (Applicable only when the IdP role is activated with at least one protocol for browser SSO, such as SAML or WS-Federation.)
3. Select a Target Session.
4. Click **Save**.

To edit target URL mappings:

1. Click **Edit** next to the Target Session. You can change the URL, Target Type and/or Target Session. (Target Type is only applicable when the IdP role is activated with at least one protocol for browser SSO, such as SAML or WS-Federation.)
2. Click **Update**.
3. Click **Save**.

To delete target URL mappings:

1. Click **Delete** next to the Target Session.
2. Click **Save**.
(Click **Cancel** to abort the deletion.)

To change the order of target URL 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 111).

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.

Field Descriptions

Field	Description
Instance Name	A descriptive name for the provisioner instance—for example, an identity management system name.
Instance ID	An internal identifier for the provisioner instance. Must be alphanumeric with no spaces.
Type	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.
Parent Instance (Optional)	A list of configured instances for this type of provisioner. If applicable, select an instance that can be used as a basis for a parent/child configuration.

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 18). 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.

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:



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 Identity Store Provisioners screen.

Extending Identity Store Provisioner Contracts for Groups

Identity Store Provisioners may be written with an option allowing administrators to add to the core group attributes the plug-in instance requires. Both the core and

extended group attributes you define here must be mapped when you configure “Write Groups” 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.

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 Group 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**.

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 423).

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 567), 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 93).

★ SP Default URLs

Enter values that affect the user's experience when executing SP-initiated SSO operations.

Provide the default URL you would like to send the user to when Single Sign On (SSO) has succeeded.

Provide the default URL you would like to send the user to when Single Logout (SLO) has succeeded.

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 15).



Note: For specific parameters required or allowed for Application Endpoints, see [“SP Endpoints”](#) on page 567.

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 576).

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 571).

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 111). You must also select the associated XASP checkbox.

Main
Attribute Requester Mapping

★ Attribute Requester Mapping

During a request as part of the X.509 Attribute Sharing Profile, the Issuer DN, if supplied, is evaluated against the entries in the first table below in hierarchical order until a match is found. Use regular expressions to match different Issuer DNs to the same IdP Connection. If a match is found, the corresponding IdP connection is selected to issue the Attribute Query. If no Issuer DN match is found or if it is not provided, the Subject DN from the request is compared against the entries in the second table in a similar manner. If no Subject DN expression matches, then the default IdP connection is selected.

ISSUER DN PATTERN	IDP CONNECTION NAME	ACTION
ou=it,o=pingidentity,c=us	Industrial_Ids	Edit / Delete
<input type="text"/>	Industrial_Ids ▾	<input type="button" value="Add"/>
SUBJECT DN PATTERN	IDP CONNECTION NAME	ACTION
*.ou=resources,dc=corp,dc=pingidentity,dc=com	Industrial_Ids	Edit / Delete
<input type="text"/>	Industrial_Ids ▾	<input type="button" value="Add"/>
DEFAULT	Industrial_Ids ▾	

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 111). These endpoints are built into PingFederate and cannot be changed.

PingFederate provides a favorite icon for all Protocol Endpoints. For more information, see [“Customizing the Favicon for Application and Protocol Endpoints”](#) on page 101.

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 63).

The table below describes each endpoint:

Table 22: PingFederate SP Endpoints

Service	URL and Description
Single Logout Service (SAML 2.0)	/sp/SLO.saml2 The URL that receives and processes logout requests and responses.
Assertion Consumer Service (SAML 2.0)	/sp/ACS.saml2 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.

Table 22: PingFederate SP Endpoints (Continued)

Service	URL and Description
Artifact Resolution Service (SAML 2.0)	<p data-bbox="915 306 1143 331">/sp/ARS.ssam12</p> <p data-bbox="915 348 1409 464">The SOAP endpoint that processes artifacts returned from a federation partner to retrieve the referenced XML message on the back channel.</p> <p data-bbox="915 480 1377 506">(See “Important” footnote in this table.)</p>
Metadata Service	<p data-bbox="915 527 932 552">/</p> <p data-bbox="915 569 1386 684">The default endpoint (empty path) from which partners can retrieve Auto-Connect metadata (see “Using Auto-Connect” on page 32).</p>
Assertion Consumer Service (SAML 1.x)	<p data-bbox="915 705 1127 730">/sp/acs.sam11</p> <p data-bbox="915 747 1398 831">A SAML 1.x implementation URL that receives and processes assertions from an IdP.</p>
Single Sign-on Service (WS-Federation)	<p data-bbox="915 856 1094 882">/sp/prp.wsf</p> <p data-bbox="915 898 1409 982">The WS-Federation implementation URL that receives and processes security tokens and SLO messages.</p>
WS-Trust STS (two endpoints)	<p data-bbox="915 1008 1094 1033">/sp/sts.wst</p> <p data-bbox="915 1050 1403 1228">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.</p> <p data-bbox="915 1245 1094 1270">/pf/sts.wst</p> <p data-bbox="915 1287 1409 1434">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 “Token Exchange Mapping” on page 155).</p> <p data-bbox="915 1451 1403 1661">Note: If multiple token-generator instances of the same type are configured for the same connection or token-to-token mapping, a query parameter, <code>TokenGeneratorId</code>, must be added to either of these endpoints—see “Configuring Token Generators” on page 509.</p> <p data-bbox="915 1677 1354 1730">(See also “Important” footnote in this table.)</p>
<p data-bbox="535 1749 1403 1873">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—<code>*.ssam1*</code> and <code>*.wst</code> (see “Changing Configuration Parameters” on page 80).</p>	



Note: For any connection using multiple virtual server IDs (see [“Multiple Virtual Server IDs”](#) on page 39), each virtual server ID has its own set of protocol endpoints. Use Export Metadata on Main Menu to export a connection metadata for your partner. For more information, see [“Exporting Metadata”](#) on page 63.

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 383. Then turn to [“WS-Trust STS Configuration”](#) on page 469.

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 37). Your federation partner must supply some connection settings and other information (see [“Configuration Data Exchange”](#) on page 40).



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 464).

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 362). In preparation for configuring a new SSO connection, you need to know which adapter instance to use (see [“Configuring Target Session Mapping”](#) on page 393). (No adapters are required for a connection that uses only the Attribute Query profile—see [“Configuring the Attribute Query Option”](#) on page 425.)

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 122).

Accessing IdP Connections

You can create, modify or import 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, import a 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 378).



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 378).

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 378).

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.

To import a connection:

1. Click **Import**.
2. In the Import Connection screen, browse to a connection XML file.
3. (Optional) Select the Allow Update checkbox. When selected, if the connection already exists, it will be overwritten.
4. Click **Import** and **Done**.

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 or import individual connection configurations.



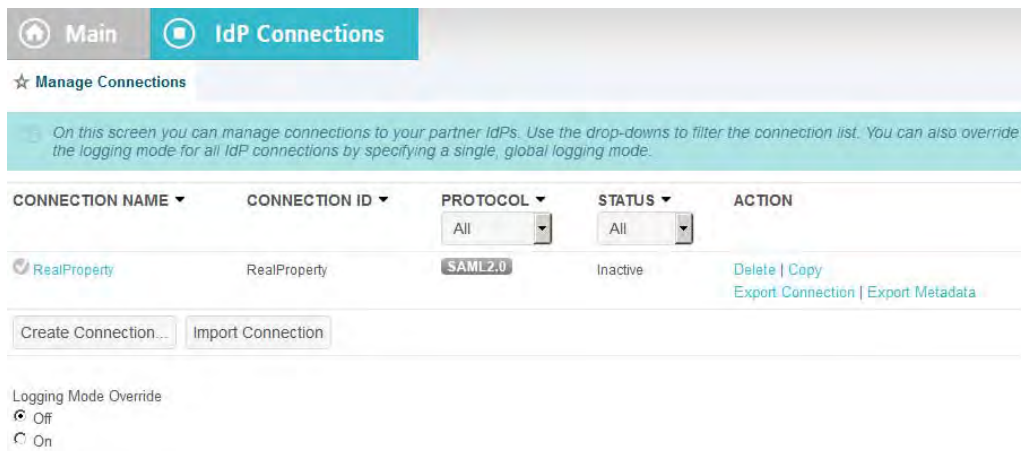
Note: The connection export function results in an XML file that you can modify and import into the same PingFederate server or another PingFederate server acting in the same federation role (IdP or SP) at your site. You can import connections in this screen. Alternatively, you can use the Connection Management Service to complete the task (see [“Importing Connections”](#) on page 593). Finally, you also have the option to automate this process (see [“Automating Configuration Migration”](#) on page 84).

- Export metadata about a connection to expedite your partner’s corresponding configuration (see [“Exporting Metadata”](#) on page 63).

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 53).



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 380.



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 376 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 383).
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.

To import a connection:

1. Click **Import**.
2. In the Import Connection screen, browse to a connection XML file.
3. (Optional) Select the Allow Update checkbox. When selected, if the connection already exists, it will be overwritten.
4. Click **Import** and **Done**.

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 63).

2. Complete the steps remaining in the Export Metadata screen flow (starting at [Step 4](#) under “[To export connection metadata:](#)” on page 64).

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.



Note: The Virtual ID displays the default virtual server ID of the connection, if any (see “[General Connection Information](#)” on page 383).

To filter the list by Protocol and/or Status:

- ▶ Select a filter criterion from either or both of the drop-down lists.

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 116).

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. You may, however, still edit the connection by clicking its name.

In addition, Action links are disabled (except for **Delete**, if the connection is Inactive and/or In Use) until the connection is validated.

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 Attribute Mapping Method”](#) on page 397). 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.

Main	IdP Connections	IdP Connection
★ Connection Type	Connection Options	General Info
<p>As an SP, you are making a connection to a partner IdP. Select the type of connection needed for this IdP: Browser SSO Profiles (for Internet SSO), WS-Trust STS (for access to identity-enabled Web Services), OAuth SAML Grant (for authenticating against the PingFederate Authorization Server), Inbound Provisioning (for integrating with SaaS partners) or all.</p>		
<input checked="" type="checkbox"/> Browser SSO Profiles	Protocol SAML 2.0	
<input type="checkbox"/> WS-Trust STS		
<input type="checkbox"/> OAuth SAML Grant		
<input type="checkbox"/> Inbound Provisioning		

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 111).



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 175).

- ▶ To configure an Inbound Provisioning connection, make that selection and choose to support provisioning of users only (User Support) or users and groups (User and Group Support).



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 111).

- ▶ (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 36).

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 111).

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 381).

- ▶ 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), PingFederate 7.3 uses this information to populate the Base URL field (see “[General Connection Information](#)” on page 383). 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 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 53).

Main
IdP Connection

Connection Type
Connection Options
Import Metadata
★ General Info
Browser SSO
Credentials
Activation & Summary

This information identifies your partner's unique connection identifier (Connection ID). Connection Name represents the plain-language identifier for this connection. Optionally, you can specify multiple virtual server IDs for your own server to use when communicating with this partner. If set, these virtual server IDs will be used in place of the unique protocol identifier configured for your server in Server Settings. The Base URL may be used to simplify configuration of partner endpoints.

Partner's Entity ID (Connection ID) *

Connection Name *

Virtual Server IDs Add

Base URL

Company

Contact Name

Contact Number

Contact Email

Error Message:

errorDetail.spSsoFailure

Logging Mode
 None
 Standard
 Enhanced
 Full

Field Descriptions

Field	Description
Partner's Entity ID/ Issuer/ Partner's Realm (Connection ID)	(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.
Connection Name	(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.
Virtual Server IDs	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).

Field	Description
Base URL	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.
Company	The name of the partner company to which you are connecting.
Contact Name	The contact person at the partner company.
Contact Number	The phone number of the contact person at the partner company.
Contact Email	The email address for the contact person at the partner company.
Error Message	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 93). 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 98. If localization is not needed, you may also specify a message directly in this field to change the default.
Logging Mode	The level of transaction logging applicable for this connection (see).

For a new connection:

- Fill in the information needed and click **Next**.

Connection ID and Connection Name are required (see "Field Descriptions" above).

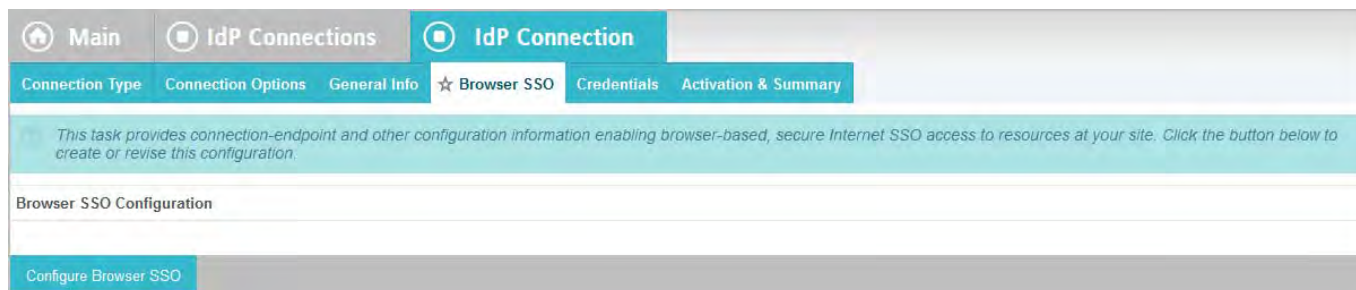
Note that the values in Virtual Server IDs identify your own federation deployment for this connection only and override the ID you specified under Server Settings (see ["Federation Server Identification"](#) on page 38).

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).



► 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 454). 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 382).



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 387
- “[User-Session Creation](#)” on page 390

- “Selecting an Identity-Mapping Method” on page 390
- “Defining an Attribute Contract” on page 392
- “Configuring Target Session Mapping” on page 393
- “Configuring SAML Protocol Settings” on page 415
 - “Specifying SSO Service URLs (SAML)” on page 416
 - “Specifying SLO Service URLs” on page 419
 - “Choosing Allowable SAML Bindings (SAML)” on page 420
 - “Setting an Artifact Lifetime (SAML 2.0)” on page 421
 - “Specifying Artifact Resolver Locations” on page 421
 - “Configuring Signature Policy” on page 423
 - “Configuring XML Encryption Policy (for SAML 2.0)” on page 424

WS-Federation SSO Steps

- “User-Session Creation” on page 390
 - “Selecting an Identity-Mapping Method” on page 390
 - “Defining an Attribute Contract” on page 392
 - “Configuring Target Session Mapping” on page 393
- “Configuring SAML Protocol Settings” on page 415
 - “Specifying a Service URL (WS-Federation)” on page 417

SAML 1.x SSO Steps

- “Choosing SAML Profiles” on page 387
- “User-Session Creation” on page 390
 - “Selecting an Identity-Mapping Method” on page 390
 - “Defining an Attribute Contract” on page 392
 - “Configuring Target Session Mapping” on page 393
- “Configuring SAML Protocol Settings” on page 415
 - “Specifying SSO Service URLs (SAML)” on page 416
 - “Choosing Allowable SAML Bindings (SAML)” on page 420
 - “Specifying Artifact Resolver Locations” on page 421
 - “Configuring Signature Policy” on page 423

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 37). 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 27).

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.

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.

If you are a federation hub, bridging an identity provider to one or more service providers, you must associate one or more connection mapping contracts to the IdP connection (see [“Federation Hub”](#) on page 42 and [“Connection Mapping Contracts”](#) on page 162 for more information).

For both scenarios, the configuration includes:

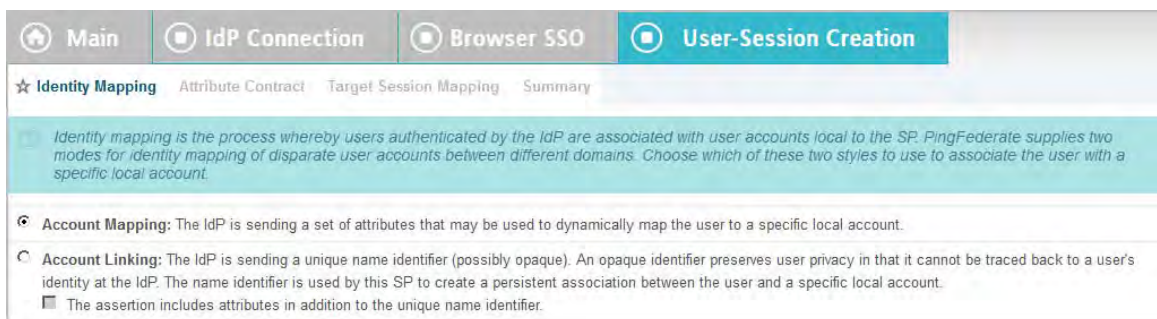
- Choosing an identity-mapping method (see [“Selecting an Identity-Mapping Method”](#) next).
- Defining the [attribute contract](#) you will use with this partner, if any (see [“Defining an Attribute Contract”](#) on page 392).
- Configuring instances of one or more target sessions (see [“Configuring Target Session Mapping”](#) on page 393) and specifying how they are used to fulfill the contract.

User-Session Configuration	
Identity Mapping	Not Configured
Attribute Contract	SAML_SUBJECT
Adapter Instances	0
Connection Contract Mappings	0


► 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 18). 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 37).




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 20).

 **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 427).

- ▶ 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 127.

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 **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 21). 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 390).

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 25).

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 **Attribute Contract** on the Summary screen.

If this step is not in the list, then you have chosen to use [account linking](#) and specified that the IdP is not including additional attributes in the assertion (see “[Selecting an Identity-Mapping Method](#)” on page 390).

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 Target Session Mapping

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 15.)

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 may also deploy an IdP connection to bridge an identity provider to one or more service providers.



Tip: In this scenario, PingFederate is a federation hub for both parties. (See “[Federation Hub](#)” on page 42 and “[Bridging multiple IdPs to an SP](#)” on page 44 for more information.)

PingFederate uses connection mapping contracts to associate this IdP connection with the applicable SP connections to the service providers (see “[Connection Mapping Contracts](#)” on page 162).

Each connection mapping contract has its own set of attributes which you map values from the assertions.

You must associate at least one target session, an adapter instance or a connection mapping contract, 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, you should map multiple adapter instances. If you are bridging an identity provider to multiple service providers, you should map multiple connection mapping contracts.

When target sessions are restricted to certain virtual server IDs, the allowed IDs are displayed under the Virtual Server IDs column.



Note: If you configure multiple target sessions for a connection, PingFederate selects the applicable adapter instance or connection mapping contract at runtime based on the target resource information in the requests and your configuration (see “[Configuring Target URL Mapping](#)” on page 366).

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 **Target Session Mapping** on the Summary screen.

To begin mapping an adapter:

- ▶ Click **Map New Adapter Instance** and follow the configuration screens (see the following sections for more information).

To begin mapping a connection mapping contract:

- ▶ Click **Map New Connection Contract Mapping** and follow the configuration screens (see the following sections for more information).

To begin modifying an existing target session mapping:

- ▶ Click the **Adapter Instance Name** or **Connection Mapping Contract Name**, and navigate through the steps to the information you want to change.

Choosing a Target Session

Use this screen to associate an SP adapter instance or a connection mapping contract with an IdP connection.



Tip: An SP adapter instance is available for use within an IdP connection only after it has been deployed and configured in PingFederate.

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 395).
 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. However, the same adapter instance may be mapped by multiple connections.



Tip: Adapter contracts for some adapters can be customized for individual connection requirements (see “[Configuring SP Adapters](#)” on page 362).

To select a connection mapping contract:

- ▶ Choose a connection mapping contract from the drop-down menu and click **Next** to continue.
 If the connection mapping contract you need is not available, click **Manage Connection Mapping Contracts** to define one or more connection mapping contracts you need for this connection.
 Note that a particular connection mapping contract can be mapped only once per connection. You may however map the same contract to multiple connections.

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 18 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:

Override Screen	Manual Section
Adapter	See “Configuring an Adapter Instance” on page 364.
Actions	See “Choosing Adapter Actions” on page 365.
Extended Contract	See “Extending Adapter Contracts” on page 365.

► To remove any adapter connection override, clear the **Override Instance Settings** checkbox, and then complete the rest of the setup.

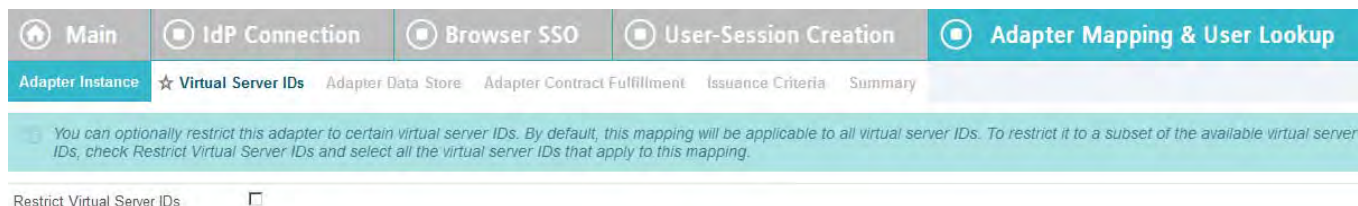
To reach this screen:

1. Click the connection name on the Main Menu.
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 Instance** on the Summary screen.
7. Select the **Override Instance Settings** checkbox.
8. Click **Next**.

Restricting a Target Session to Certain Virtual Server IDs

When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), you have the option to enforce authentication requirements by restricting a target session (an adapter or a connection mapping contract) to certain virtual server IDs. This optional setting can

be applied to each target session added to the connection using virtual server IDs. By default, no restriction is imposed.



To restrict a target session 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 virtual server IDs that you want to allow for this target session.
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 **Target Session Mapping** on the Summary screen.
7. Click the target session 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 38.

Selecting an Attribute Mapping Method

To populate the attributes required by the target session (an adapter contract or a connection mapping 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 122).



- ▶ 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 Target Session Fulfillment”](#) on page 406).



Tip: To determine whether you need to look up additional values, compare your attribute contract against the contract of your target session (see [“Defining an Attribute Contract”](#) on page 392 and [“Choosing a Target Session”](#) on page 394). If the target session 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 Target Session Fulfillment”](#) on page 406.)

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 398.
2. See the following sections in this manual, depending on the type of data store:

Data Store Type	Related Manual Section
JDBC	<ul style="list-style-type: none"> • “Selecting a Database Table and Columns” on page 399 • “Configuring a Database Filter” on page 401
LDAP	<ul style="list-style-type: none"> • “Configuring a Directory Search” on page 402 • “Specifying an LDAP Filter” on page 404
Custom	<ul style="list-style-type: none"> • “Configuring Custom Data-Source Filters” on page 406 • “Selecting Custom Data-Source Fields” on page 406

3. See [“Configuring Target Session Fulfillment”](#) on page 406.

Choosing a Data Store

This screen allows you to choose a data store from a previously configured list (see [“Managing Data Stores”](#) on page 122). 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 or the connection mapping contract (see [“Configuring Target Session Mapping”](#) on page 393).

The screenshot shows a web interface with a navigation bar at the top containing: Main, IdP Connections, IdP Connection, Browser SSO, and User-Session Creation. Below the navigation bar are tabs: Adapter Instance, Adapter Data Store, Data Store (selected), and Summary. A teal banner contains the text: "Please select the data store from which local data, along with the attributes supplied in the SAML assertion, will be used to fulfill the SP Adapter Contract." Below this is a form with two fields: "Active Data Store" with a dropdown menu showing "- SELECT -" and a small square icon to its right, and "Data Store Type" with the value "None". At the bottom left of the form is a button labeled "Manage Data Stores...".

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session 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 399).

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 122).

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 401).



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

and

dev.mysql.com/doc/refman/5.1/en/option-files.html

Field Descriptions

Field	Description
Schema	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.
Table	Displays the table(s) contained in the database. Select the table to retrieve data from the data store.
Columns to return from SELECT	Displays the columns available from the selected table. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session 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 398.

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.

- 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 target session.

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 285).

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 399).

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session 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 398.

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 Attribute Mapping Method”](#) on page 397).

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 target session (an SP adapter or a connection mapping contract) in order to access a resource on your system.

Field Descriptions

Field	Description
Base DN	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.
Search Scope	Determines the node depth of the query. Select Subtree, One level or Object.
Root Object Class	Specifies the object type within the LDAP hierarchy from which attributes will be returned.
Attribute	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.

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session name.
8. Click **LDAP Directory Search** on the Summary screen.

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 398).

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.



Note: When connecting to Microsoft Active Directory, if you choose the memberOf attribute, an optional checkbox, Nested Groups, appears on the right. Select this checkbox if you want PingFederate to query for groups the end users belong to directly as well as indirectly through nested group membership (if any) under the Base DN.

For example, suppose you have three groups under the Base DN, namely Canada, Washington and Seattle. Seattle is a member of Washington. Ana Smith is an end user and a member of Seattle. If the Nested Groups checkbox is selected, when PingFederate queries for Ana's memberOf attribute values, the expected results are Seattle and Washington. (When the Nested Groups checkbox is not selected (the default), the expected result is Seattle.)

For Oracle Directory Server, choose isMemberOf under Attribute for nested group membership. For more information, see [documentation about isMemberOf from Oracle](http://docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm) (docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm).

5. Repeat the last step for other attributes as needed.



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 target session.

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*})

The screenshot shows the configuration interface for the LDAP Filter. The top navigation bar includes tabs for Main, IdP Connections, IdP Connection, Browser SSO, and User-Session Creation. Below this, a sub-navigation bar highlights the LDAP Filter tab, with other options like Adapter Instance, Adapter Data Store, Data Store, LDAP Directory Search, Adapter Contract Fulfillment, Issuance Criteria, and Summary. The main content area has a light blue header with the text "Please filter the data from your directory." Below this is a "Filter" section with a text input field. Underneath is an "Assertion Values" section with a list of attributes: \${email}, \${fname}, \${lname}, and \${SAML_SUBJECT}. At the bottom, there is a link that says "View List of Available LDAP Attributes".

The left-side variable is an attribute from the data store (see [“Configuring a Directory Search”](#) on page 402).

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:

```
(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

Field	Description
Filter	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.

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session 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 398).

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 127).

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 target session (an adapter contract or a connection mapping contract). These choices are supplied by the driver implementation. Select only those needed to fulfill the target session.

Configuring Target Session Fulfillment

The last step in configuring a target session is to map each attribute required to a value (see [“Selecting an Attribute Mapping Method”](#) on page 397). If you integrate your applications with PingFederate through an SP adapter, 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. If you are bridging an identity provider to a service provider, these are the values that will be used to fulfill the connection mapping contract, which in turn will be used by the SP connection to create the assertions for the service provider (see [“Federation Hub”](#) on page 42 for more information).

Map attributes from one of these Sources:

- Account Link

This source appears only if you have elected to use [account linking](#) for an SP adapter (see [“Selecting an Identity-Mapping Method”](#) on page 390). 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. This source is not applicable to connection mapping contracts.

- 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 392).

- 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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 399, [“Configuring a Directory Search”](#) on page 402, or [“Selecting Custom Data-Source Fields”](#) on page 406).

- 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 613). 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 563).

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.

ADAPTER CONTRACT	SOURCE	VALUE	ACTIONS
E-mail Address	Assertion	email	None available
First Name	Assertion	fname	None available
Last Name	Assertion	lname	None available
Net Worth	Text	\$1,000,000	None available
subject	Assertion	SAML_SUBJECT	None available
userid	Assertion	SAML_SUBJECT	None available

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 **Target Session Mapping** under the User-Session Creation tab.
7. Click the target session 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 or connection mapping contract (see [“About Token Authorization”](#) on page 25). This token authorization can be used to restrict who can access protected resources.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT -	- SELECT -	- SELECT -			Add

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.

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 93).



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 613).



Important: When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), 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 616).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618. 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**.

Configuring OAuth Attribute Mapping

This configuration allows administrators to use assertion attributes for mapping values into OAuth persistent-grant `USER_KEY` and extended attributes (see “About OAuth” on page 10).



Note: This option is presented only when enabled for a connection (see “Choosing IdP Connection Options” on page 382).

- ▶ 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 216.

- ▶ 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), the extended attributes (to fulfill

the access token), and `USER_NAME` (the name shown to end-users on permissions screens) for persistent access-token grants.



Note: The `USER_KEY` values must be unique across all end users because it is the identifier to store and to retrieve persistent grants. If `SAML_SUBJECT` uniquely identifies all end users, you can map `SAML_SUBJECT` to `USER_KEY`.

Main IdP Connections IdP Connection Browser SSO			
Data Store Database Table and Columns Database Filter Contract Fulfillment Issuance Criteria Summary			
Manage attribute mappings into the <code>USER_KEY</code> value of persistent grants and the <code>USER_NAME</code> value displayed to end users.			
CONTRACT	SOURCE	VALUE	ACTIONS
USER_KEY	Assertion	- SELECT -	None available
USER_NAME	JDBC	- SELECT -	None available

Map each attribute 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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- 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 399 or [“Configuring a Database Filter”](#) on page 401).
- 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 613). 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:

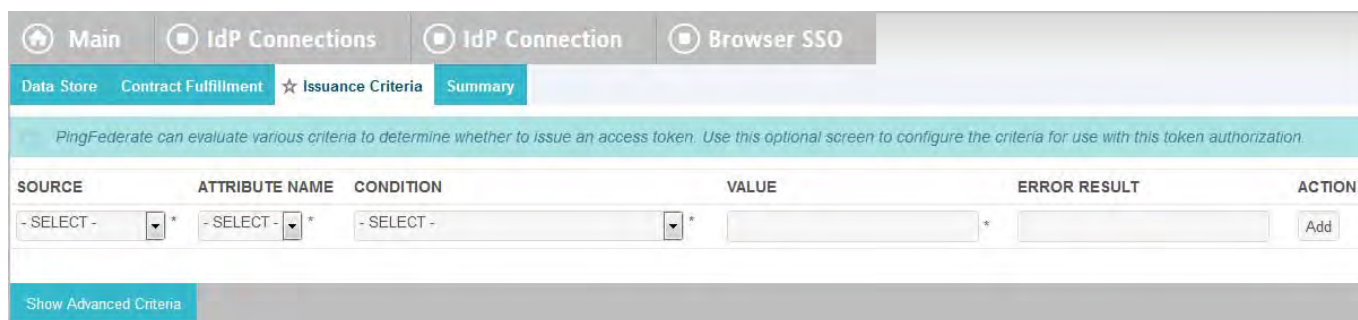
1. Click the connection name on the Main Menu.
 Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **Browser SSO** under the IdP Connection tab.
3. Click **Configure Browser SSO**.
4. Click **OAuth Attribute Mapping** under the Browser SSO tab.
5. Click **Configure OAuth Attribute Mapping**.
6. Click **Contract Fulfillment** on the Summary screen.

To map attributes:

1. Choose a Source for each attribute.
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 25). 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.

- 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.
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 93).



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 613).



Important: When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), 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 616).

- Use the in-line editor box to enter the OGNL expression. For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618. 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 387). The screen also displays configured information.

(For WS-Federation, the configuration of bindings is not applicable.)

Protocol Settings Configuration	
Outbound SSO Bindings	POST
Outbound SLO Bindings	POST
Inbound Bindings	POST
Signature Policy	SAML-standard
Encryption Policy	No Encryption

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 389).

Home Main IdP Connections IdP Connection Browser SSO Protocol Settings		
★ SSO Service URLs SLO Service URLs Allowable SAML Bindings Signature Policy Encryption Policy Summary		
As the SP, you send authentication requests (AuthnRequests) for single sign-on to the IdP's SSO Service. Depending on the situation, the IdP may have several endpoints available. Please provide the endpoints that you want to use when sending these requests.		
BINDING	ENDPOINT URL	ACTION
POST	/idp/SSO.saml2	Edit / Delete
- SELECT - *	<input type="text"/>	<input type="button" value="Add"/>

Field Descriptions

Field	Description
Binding (SAML 2.0)	The transport type agreed upon by you and your partner: Artifact, POST, or Redirect.
Endpoint URL	The location where your IdP receives SSO messages.

To reach this screen for editing:

- Click a connection name on the Main Menu.
Click **Manage All IdP**, if needed, to see a full list of connections.
- Click **Browser SSO** under the IdP Connection tab.
- Click **Configure Browser SSO**.
- Click **Protocol Settings** on the Summary screen.
- Click **Configure Protocol Settings**.
- 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 387).

To define an Endpoint URL:

- If you are using SAML 2.0, select the Binding your partner specifies for the Endpoint.
- Enter the fully qualified Endpoint URL or a relative path if you have defined a base URL (see [“General Connection Information”](#) on page 383).
- If you are using SAML 2.0, click **Add**.
- 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.

To protect against session token hijacking, PingFederate provides an option to validate wreply for SLO. When the option is enabled, you can specify additional allowed domains and paths in this screen. PingFederate validates the locations against a consolidated list of allowed domains and paths from all active WS-Federation connections before redirecting the end users to their destinations.

Main		IdP Connection		Browser SSO		Protocol Settings	
☆ Service URL Default Target URL Signature Policy Summary							
<i>As the SP you send security token requests and SLO messages to the IdP. Specify here the URL where the IdP is listening for these messages.</i>							
Endpoint URL <input type="text"/>							
For incoming SLO cleanup messages, your partner may specify a redirect location via the wreply parameter. You can define allowed values for this parameter below.							
Require HTTPS	Valid Domain Name (leading wildcard "*", allowed)	Valid Path (leave blank to allow any path)	Allow Any Query/Fragment	Action			
<input checked="" type="checkbox"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="button" value="Add"/>			



Note: Settings to enter additional allowed domains and paths appear only if the option to validate wreply for SLO is enabled (see [“Managing Partner Redirect Validation”](#) on page 145).

To define a service URL:

- ▶ Enter the fully qualified URL or a relative path if you have defined a base URL (see [“General Connection Information”](#) on page 383). You must include the initial slash if you are entering only a relative path.

To specify additional allowed domains and paths:

1. Indicate whether to require HTTPS.



Important: This selection is recommended to ensure that the validation will always prevent message interception for this type of potential attack, under all conceivable permutations.

2. Enter the expected domain or IP address under Valid Domain Name.

Use the domain only, without qualifiers. For example:

`company.com`

Using an initial wildcard and period for a domain name will cover multiple subdomains. For example:

`*.company.com`

covers `hr.company.com` or `email.company.com`.

(Optional) Enter the exact path of the resource (case-sensitive) under Valid Path. 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:

/app/Consumer.jsp

allows /app/Consumer.jsp but rejects /app/consumer.jsp.



Tip: You can also enter multiple query parameters with or without a fragment.

For example:

/app/Consumer.jsp?area=West&team=IT#ref1001

matches /app/Consumer.jsp?area=West&team=IT#ref1001 but not /app/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 resource.



Note: When selected, no query parameter and/or fragment is allowed in the path.

Click **Add**.

3. Repeat the previous step to add additional entries as needed.
4. Click **Save**.

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 387).

Field Descriptions

Field	Description
Binding	The transport type agreed upon by you and your partner: Artifact, POST, Redirect, or SOAP.
Endpoint URL	A location where your IdP receives SLO request messages.
Response URL	(Optional) A location where the IdP receives SLO logout response messages. Use this endpoint when you are part of a chain of session participants. When omitted, the PingFederate SP server sends logout responses to the Endpoint URL.

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** on the Summary screen.
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 383).
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 387). You and your partner can agree to standardize on one binding type or select different bindings for different profile scenarios.

The screenshot shows the 'Protocol Settings' tab selected in the navigation bar. Below it, the 'Allowable SAML Bindings' sub-tab is active. The main content area displays a question: 'When the IdP sends messages, over what SAML bindings do you want to receive them?'. Below the question is a list of binding options with checkboxes: 'Artifact' (unchecked), 'POST' (checked), 'Redirect' (unchecked), and 'SOAP' (unchecked).

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** on the Summary screen.
5. Click **Configure Protocol Settings**.
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.

The screenshot shows a web interface with a navigation bar at the top containing tabs: Main, IdP Connection, Browser SSO, Protocol Settings (selected), and Summary. Below the navigation bar is a sub-menu with options: SSO Service URLs, SLO Service URLs, Allowable SAML Bindings, Artifact Lifetime (selected), Artifact Resolver Locations, Signature Policy, Encryption Policy, and Summary. A light blue banner contains the text: "Artifacts are meant to be short-lived tokens representing an issued message. For how long should the recipient of the artifact be allowed to retrieve the corresponding message?". Below this banner is a form field labeled "Artifact Lifetime" with a text input containing the value "60" and a unit label "second(s) *".

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 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** on the Summary screen.
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.

Please provide the remote party URLs that you will use to resolve/translate the artifact and get the actual protocol message.

INDEX	URL	ACTION
0	/idp/ARS.ssaml2	Edit / Delete
<input type="text"/>	<input type="text"/> *	Add

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**.
4. Click **Protocol Settings** on the Summary screen.
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 383).
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 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 383).
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 372).



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 563), any URL specified in the Default Target URL field on this screen will not be used for those transactions.

The screenshot shows the 'Protocol Settings' tab selected in the top navigation bar. Below it, several sub-tabs are visible: 'SSO Service URLs', 'SLO Service URLs', 'Allowable SAML Bindings', 'Artifact Resolver Locations', 'Default Target URL' (which is highlighted with a star icon), and 'Signature Policy'. Under the 'Default Target URL' sub-tab, there is a text input field labeled 'Default Target URL'. A light blue informational banner above the field states: 'Optionally, you can specify a default target URL for this IdP connection. Entering a URL in the Default Target URL field overrides the SP Default URL SSO setting.'

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 29).

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 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.)

The screenshot shows the 'Signature Policy' sub-tab selected in the top navigation bar. A light blue informational banner at the top reads: 'Additional guarantees of authenticity may be agreed upon between you and your partner. For SP-initiated SSO, you can choose to sign authentication requests sent via the POST or redirect bindings. You can also choose to require signed assertions, regardless of the binding used.' Below this, the text says 'Specify how message authenticity and integrity is ensured:'. There are three radio button options: 'Use SAML-standard signature requirements' (which is selected), 'Specify additional signature requirements', and 'Require signed SAML Assertions (rather than signed Responses — Assertions are contained inside SAML Responses)'. Under the 'Specify additional signature requirements' option, there are two checkboxes: 'Sign AuthN requests sent over POST and Redirect bindings' and 'Require signed SAML Assertions (rather than signed Responses — Assertions are contained inside SAML Responses)'.

► 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 463).

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 463).

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 382). 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*).

Specify the URL at your IdP partner's site where attribute queries are to be sent.

Attribute Authority Service URL

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 383).

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.

Attributes requested by your application may not match exactly the attributes supplied by the IdP. Specify the mapping between these sets of attributes.

LOCAL NAME	REMOTE NAME	ACTION
EmailAddress	email	Edit / Delete
FirstName	fname	Edit / Delete
LastName	lname	Edit / Delete
<input type="text"/>	<input type="text"/>	<input type="button" value="Add"/>

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 454).

This screen also allows you to mask incoming attribute values in log files (see [“Attribute Masking”](#) on page 25). 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.

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 36). An SP can also use the feature to update existing user records.

The screenshot shows the administrative console for IdP Connection. The 'JIT Provisioning' tab is selected. Below the navigation tabs, there is a section titled 'Specify how and when to provision user accounts.' followed by a table for 'JIT Provisioning Configuration'.

JIT Provisioning Configuration	
User Attributes	email, fname, lname, SAML_SUBJECT
User Repository	None
Attribute Query	None
SQL Method	Table
Event Trigger	Only SAML Assertions containing a new user id
Error Handling	Not Configured

At the bottom of the configuration area, there is a button labeled 'Configure JIT 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 382).

- ▶ 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 392).



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.

The screenshot shows the 'User Attributes' configuration page under the 'JIT Provisioning' tab. A message states: 'User accounts are provisioned with attributes from the SAML Assertion by default. You can also retrieve additional attributes from the IdP using the Attribute Query profile.' Below this is the 'ATTRIBUTE CONTRACT' section with a list of attributes: email, fname, lname, and SAML_SUBJECT. At the bottom, there are two radio button options: 'Use only these user attributes' (which is selected) and 'Issue an Attribute Query back to the IdP to retrieve additional user attributes'.

- ▶ 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 425).

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 122).
- ▶ 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 430
 - [“Identifying Provisioning Attributes for LDAP”](#) on page 431
- ▶ If you are using MSSQL, skip to this section:
 - [“Selecting a SQL Method”](#) on page 431

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 429).

Field Description

Field	Description
Base DN	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.

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 429).

The screenshot shows the 'JIT Provisioning' configuration page. The navigation tabs include 'Main', 'IdP Connections', 'IdP Connection', and 'JIT Provisioning'. Under 'JIT Provisioning', there are sub-tabs: 'User Attributes', 'User Repository', 'Location', 'Unique ID', 'Attributes', 'Attribute Fulfillment', 'Event Trigger', and 'Error Handling'. The 'Unique ID' tab is active. Below the sub-tabs is a 'Summary' section with a teal background and the text: 'Using attributes from the Assertion, specify an expression that results in a unique user identifier, when combined with the Base DN.' Below this is a 'Filter:' label and a text input field. Underneath the input field is a section titled 'JIT Attributes' with a list of attributes: '\${Email Address}', '\${First Name}', '\${Last Name}', '\${Member Status}', and '\${SAML_SUBJECT}'. At the bottom left of the form is a link: 'View List of Available LDAP Attributes'.

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 404).

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 392). 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 434).



Note: This screen appears only for LDAP data stores (see “Identifying the User Repository” on page 429).

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.

Indicate whether you want to map attributes from the assertion directly to database table columns or to stored-procedure parameters.

SQL METHOD

Table

Stored Procedure



Note: The SQL Method screen appears only for JDBC data stores (see “Identifying the User Repository” on page 429).

- ▶ 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:

- ▶ 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 433
- ▶ If you are using a Stored Procedure, skip to this section:
 - “Specifying a Stored-Procedure Location” on page 433

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”).

Specify where user records are located in the repository.

Schema:

Table:

Columns to fulfill

EMAIL	nvarchar
FIRSTNAME	varchar
LASTNAME	varchar
USERID	varchar
EMAILADDRESS	varchar

[View Attribute Contract](#)

Field Descriptions

Field	Description
Schema	Select the table structures that store information within the database.
Table	Select the name of the database table that contains user records.
Columns to fulfill when provisioning	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 434).

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 431).

- ▶ 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 431).

Field Descriptions

Field	Description
Schema	Select the table structure that contains stored procedures within the database.
Stored Procedure	Select the stored procedure needed to provision the user database.
Procedure parameters to fulfill	For the selected procedure, all parameters and their data types are displayed. You map assertion values to the parameters on the next screen.

- ▶ 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 431). 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 613). Note that expressions must be enabled to use the `PingFederate` conversion method—see [“Enabling and Disabling Expressions”](#) on page 613.



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 392).
- 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 428).
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 613)

This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see [“Using Attribute](#)

Mapping Expressions” on page 613). 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.

- 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.

- 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.

- 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 431), because the procedure is always called for all assertions. The procedure should handle both provisioning new users and updating existing ones (if desired).

Home	Main	IdP Connection	JIT Provisioning	
User Attributes	User Repository	Location	Unique ID	Attributes
Attribute Fulfillment	★ Event Trigger	Error Handling	Summary	

Provisioning can occur as part of assertion processing for new users only, or for all users arriving from this IdP partner.

Specify the trigger that initiates a user-provisioning event:

Only SAML Assertions containing a new user id

All SAML Assertions

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 Target Session Mapping”](#) on page 393).

Home	Main	IdP Connection	JIT Provisioning	
User Attributes	User Repository	Location	Unique ID	Attributes
Attribute Fulfillment	Event Trigger	★ Error Handling	Summary	

Depending upon the nature of the target application and the agreement with the partner, specify the server's behavior when a provisioning request fails for any reason.

How should the server handle a failure in a user provisioning request?

Send the user's attributes to the target application

Abort the SSO transaction

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.

- ▶ When you are finished with a new configuration, click **Done** and then **Save** on the JIT Provisioning screen.

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 36).

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 441). 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 444).

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 613).



Note: SCIM-client requests must include authentication credentials, which you configure later (see [“Configuring Security Credentials”](#) on page 454). 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.

The screenshot shows the administrative console interface for configuring an IdP connection. The breadcrumb trail is: Main > IdP Connection > Inbound Provisioning. The page title is "Specify how and when to provision user accounts and groups." Below this, there is a section for "Inbound Provisioning Configuration" with a "Repository" field set to "None". At the bottom, there is a "Configure Inbound Provisioning" button.

- ▶ To continue, click **Configure Inbound Provisioning**.

Specifying the User Repository

PingFederate currently supports AD user stores (requires an LDAPS connection to the data 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 122).

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 368).

Identifying an LDAP User-Record Location

After choosing a data store, indicate where in the data store user and group records exist so PingFederate can create, read, update, or delete/disable them.



Note: This screen appears only if you are configuring an LDAP user store for provisioning.

To specify a base DN:

- ▶ 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.

Main IdP Connection Inbound Provisioning

Repository Location Unique User ID Unique Group ID Write Users Read Users Delete/Disable Users Write Groups Read Groups Summary

Using attributes from the SCIM request, specify an expression that results in a unique user identifier, when combined with the Base DN.

Filter

CN=\${userName}

SCIM Attributes

- \${active}
- \${addresses.home.country}
- \${addresses.home.formatted}
- \${addresses.home.isPrimary}
- \${addresses.home.locality}



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 404).

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**.


Defining a Unique Group ID

On the Unique Group ID screen, create an LDAP filter to resolve groups for SCIM operations. PingFederate uses this expression in conjunction with the Base DN (see the previous section) to add new groups.


 **Note:** This screen appears only if you are configuring an LDAP user store for provisioning and the User and Group Support checkbox is selected in the Connection Type screen (see [“Choosing an IdP Connection Type”](#) on page 381).

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 404).

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.

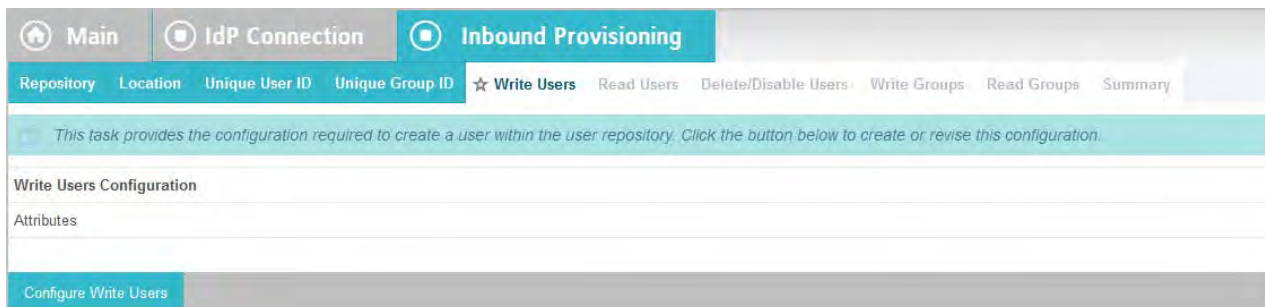
 **Note:** If you are unfamiliar with writing LDAP queries, please refer to the documentation accompanying your LDAP installation.

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

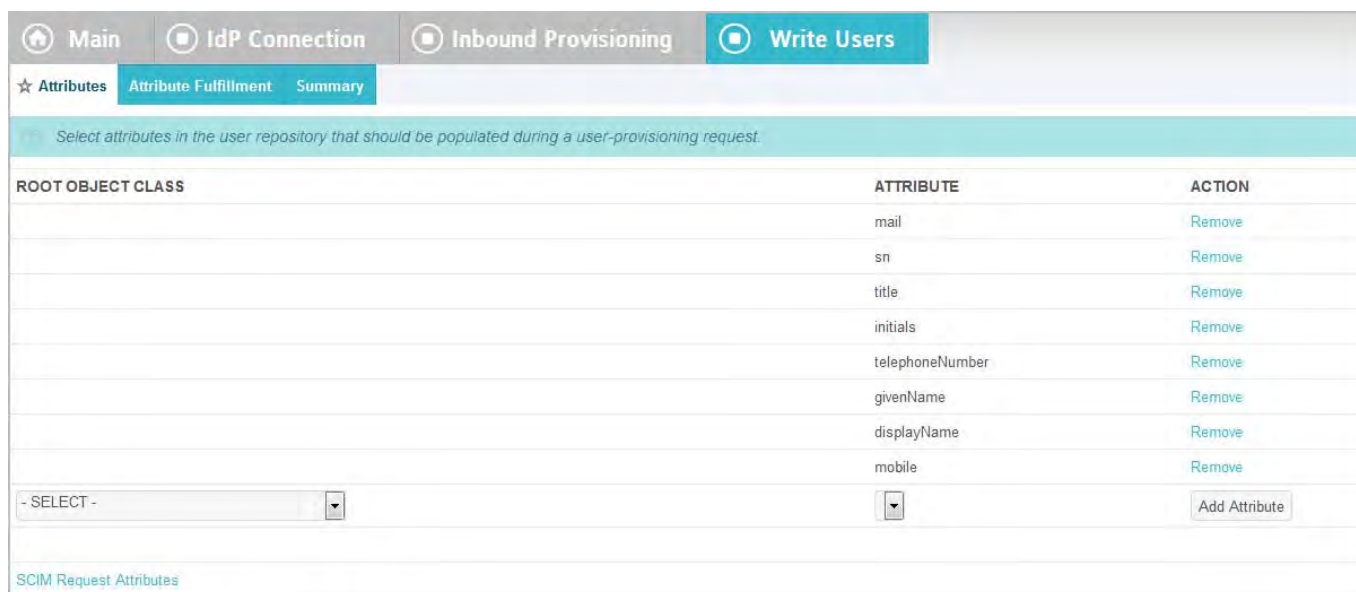
To configure how PingFederate completes create and update operations for user accounts from a SCIM request, identify incoming attributes and map them to data-store attributes.



► 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.



Note: This screen appears only if you are configuring an LDAP user store for provisioning.

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.

The screenshot shows the 'Write Users' configuration page. At the top, there are navigation tabs: Main, IdP Connections, IdP Connection, Inbound Provisioning, and Write Users (which is active). Below the tabs are sub-tabs: Attributes, Attribute Fulfillment (selected), and Summary. A teal banner contains the instruction: 'Map attribute Sources and Values for each repository attribute that should be populated.' Below this is a table with the following structure:

TARGET ATTRIBUTE	SOURCE	VALUE	ACTIONS
displayName	- SELECT -		None available
givenName	- SELECT -		None available
homePhone	- SELECT -		None available
initials	- SELECT -		None available
mail	- SELECT -		None available
mobile	- SELECT -		None available
sn	- SELECT -		None available
telephoneNumber	- SELECT -		None available
title	- SELECT -		None available
url	- SELECT -		None available

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- Expression (when enabled—see [“Enabling and Disabling Expressions”](#) on page 613)

This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see [“Using Attribute Mapping Expressions”](#) on page 613). 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 User**
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.

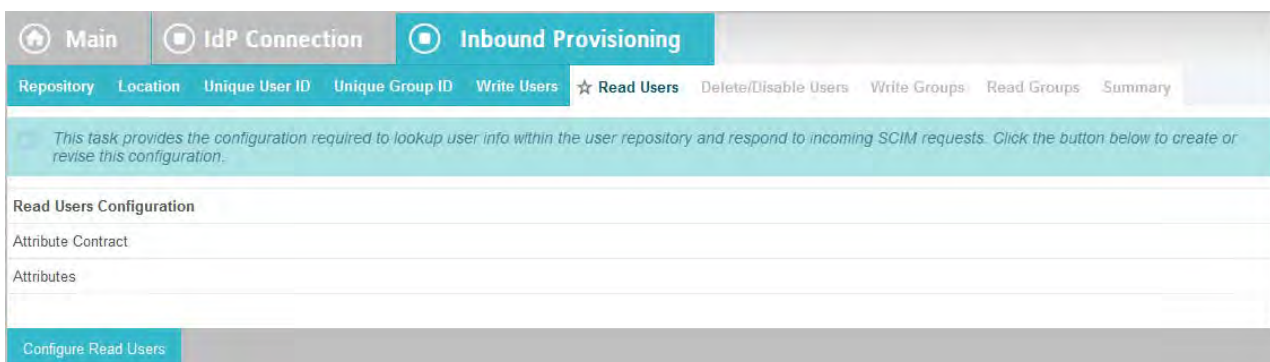
- When you finish with a new configuration, click **Done**.

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.

☆ Attribute Contract | Attributes | Attribute Fulfillment | Summary

An Attribute Contract is a set of user attributes that you will send to your partner in the SCIM response.

ATTRIBUTE CONTRACT

userType

EXTEND THE CONTRACT	MASK VALUES IN LOG	ACTION
<input type="text"/>	<input type="checkbox"/>	Add

[Available SCIM attributes](#)

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
- 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:

- 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.

Home Main IdP Connection Inbound Provisioning **Read Users**

Attribute Contract ★ Attributes Attribute Fulfillment Summary

Select attributes in the user repository that should be used to populate a response to a user-provisioning request.

ROOT OBJECT CLASS	ATTRIBUTE	ACTION
	mail	Remove
	title	Remove
	sn	Remove
	initials	Remove
	telephoneNumber	Remove
	givenName	Remove
	displayName	Remove
	mobile	Remove
- SELECT -		Add Attribute

SCIM Request Attributes



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.

Home Main IdP Connection Inbound Provisioning **Read Users**

Attribute Contract Attributes ★ Attribute Fulfillment Summary

Map attribute Sources and Values for each SCIM attribute that should be populated.

TARGET ATTRIBUTE	SOURCE	VALUE	ACTIONS
displayName	LDAP	displayName	None available
emails.work.value	LDAP	mail	None available
name.familyName	LDAP	sn	None available
name.givenName	LDAP	givenName	None available
name.middleName	LDAP	initials	None available
phoneNumbers.mobile.value	LDAP	mobile	None available
phoneNumbers.work.value	LDAP	telephoneNumber	None available
title	LDAP	title	None available

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- Expression (when enabled—see [“Enabling and Disabling Expressions”](#) on page 613)

This option provides more complex mapping capabilities—for example, transforming outgoing values into different formats (see [“Using Attribute Mapping Expressions”](#) on page 613). 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.

- 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.

- 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.
- Click **Save** on the Inbound Provisioning screen.

Handling SCIM Delete Requests

Use this screen to define how SCIM delete requests for user accounts are handled within your user data store.



Important: If the group support option is enabled (see “[Choosing an IdP Connection Type](#)” on page 381), when PingFederate receives a SCIM delete request for a group, it always removes the specified group from the data store.

The screenshot shows the 'Inbound Provisioning' configuration screen. The breadcrumb trail is: Main > IdP Connection > Inbound Provisioning. The sub-menu items are: Repository, Location, Unique User ID, Unique Group ID, Write Users, Read Users, Delete/Disable Users (highlighted), Write Groups, Read Groups, and Summary. A teal banner contains the text: 'As a SCIM Service Provider, you can define how SCIM DELETE requests are handled within your user repository. Please define how SCIM DELETE requests should be handled.' Below this, the 'SCIM DELETE message behavior' section has two radio button options: 'Disable User' (selected) and 'Permanently Delete User'.



Note: This screen appears only if you are configuring an LDAP user store for provisioning.

- ▶ 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](#)).

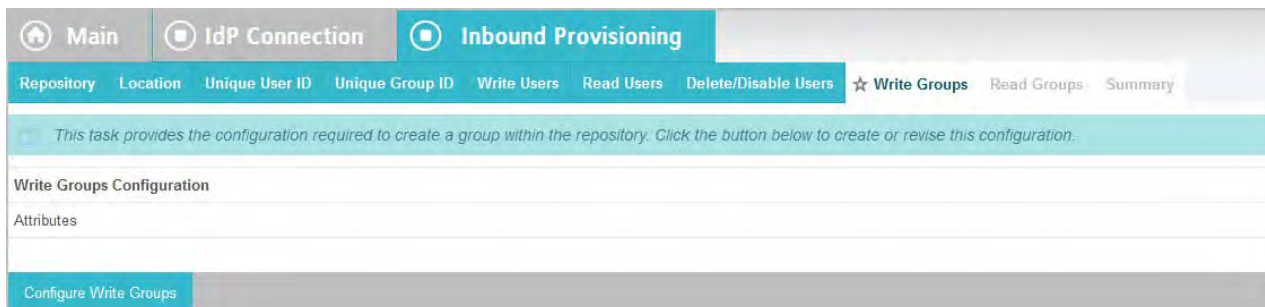


Caution: If the user is disabled through another method, PingFederate still treats that user as if they have been deleted and returns HTTP 404 response codes for all subsequent requests.

- ▶ Select **Permanently Delete User** to remove the user from the data store.

Writing Group Information to the Data Store

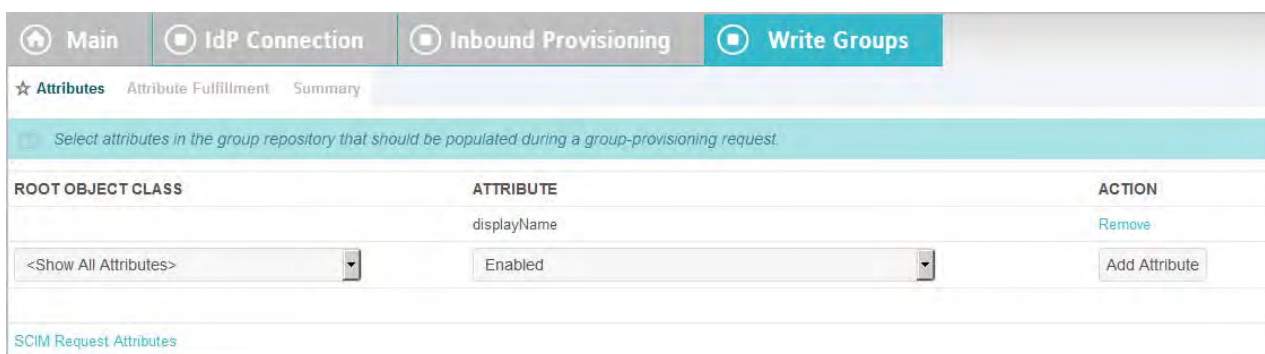
To configure how PingFederate completes create and update operations for groups from a SCIM request, identify incoming attributes and map them to data-store attributes.



► Click **Configure Write Groups** to continue.

Identifying Inbound Provisioning Group Attributes for LDAP

On the Attributes screen, select the data-store attributes you want to provision.



Note: This screen appears only if you are configuring an LDAP user store for provisioning and the User and Group Support checkbox is selected in the Connection Type screen (see [“Choosing an IdP Connection Type”](#) on page 381).

System Managed LDAP Group Attributes

Several attributes are managed internally by PingFederate and do not require mapping:

- objectClass
- objectGUID
- member

You can override the internal management of `objectClass` by selecting and mapping it to a SCIM attribute on the Attribute Fulfillment screen. In this case, the values you supply are used. The `objectGUID` and `member` 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 Groups

Use this screen to map attribute values in the SCIM request to group attributes.

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- Expression (when enabled—see [“Enabling and Disabling Expressions”](#) on page 613)
This option provides more complex mapping capabilities—for example, transforming incoming values into different formats (see [“Using Attribute Mapping Expressions”](#) on page 613). 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 Group
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 Groups Summary Screen

The Summary screen provides an overview of the inbound provisioning configuration for request mapping.

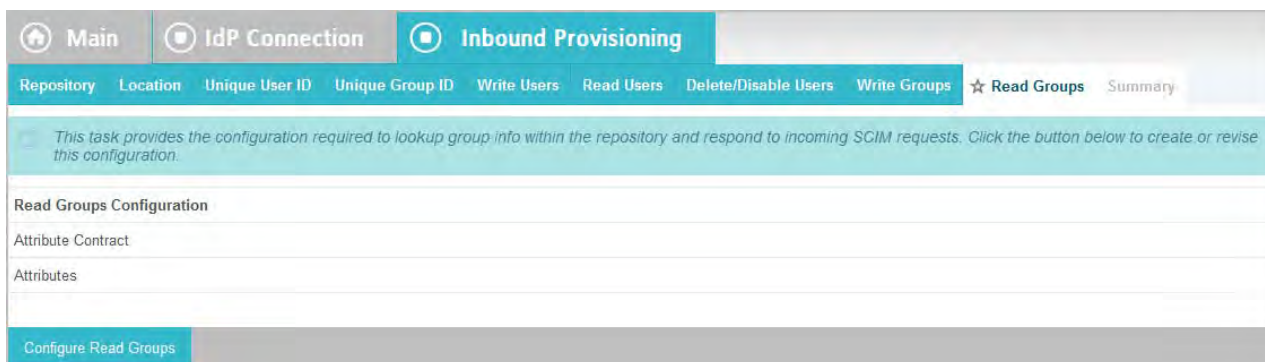
- ▶ When you finish with a new configuration, click **Done**.

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 Write Groups screen.
3. Click **Next** on the Inbound Provisioning screen.

Configuring a SCIM Response for Groups

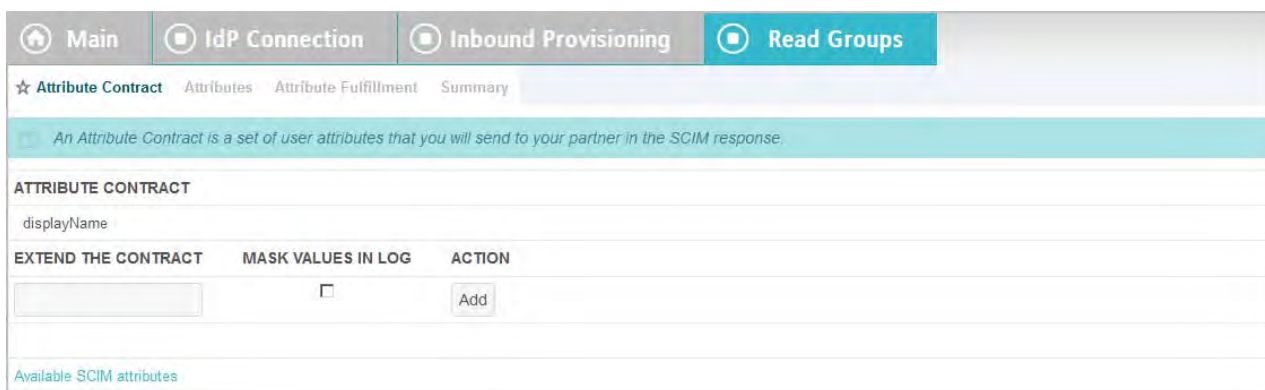
To configure a SCIM response to a request to read and return provisioned SCIM attributes, identify and map the group attributes you want to include.



► Click **Configure Read Groups** to continue.

Identifying Expected Group Attributes for the SCIM Response

An attribute contract is a set of group attributes that you and your partner have agreed will be sent in a SCIM response for this connection. The attributes you mapped to group attributes in the Write Groups 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 Group Attributes

There are several SCIM attributes that are managed internally by PingFederate and are unavailable for inclusion in the attribute contract:

- id

- members

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:

- ▶ Click **Delete** under Action for the attribute.

Identifying LDAP Group Attributes for the SCIM Response

Select the LDAP attributes you want to map to attributes in the SCIM response.

ROOT OBJECT CLASS	ATTRIBUTE	ACTION
	displayName	Remove

<Show All Attributes> Enabled Add Attribute

SCIM Request Attributes



Note: This screen appears only if you are configuring an LDAP user store for provisioning and the User and Group Support checkbox is selected in the Connection Type screen (see ["Choosing an IdP Connection Type"](#) on page 381).

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 Group Attributes to SCIM Response Attributes

Use this screen to map outgoing group attributes to SCIM responses to READ requests.

Main		IdP Connection		Inbound Provisioning		Read Groups	
Attribute Contract		Attributes		★ Attribute Fulfillment		Summary	
Map attribute Sources and Values for each SCIM group attribute that should be populated.							
TARGET ATTRIBUTE	SOURCE	VALUE	ACTIONS				
displayName	LDAP	displayName	None available				

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 618). (If the Expression selection is not listed, then the feature is not enabled—see [“Enabling and Disabling Expressions”](#) on page 613. For syntax and examples, see sections under [“Constructing Expressions”](#) on page 614.)

- Expression (when enabled—see [“Enabling and Disabling Expressions”](#) on page 613)
This option provides more complex mapping capabilities—for example, transforming outgoing values into different formats (see [“Using Attribute Mapping Expressions”](#) on page 613). 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.
- 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 Groups 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 Groups screen.

3. Click **Save** on the Inbound Provisioning screen.

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.



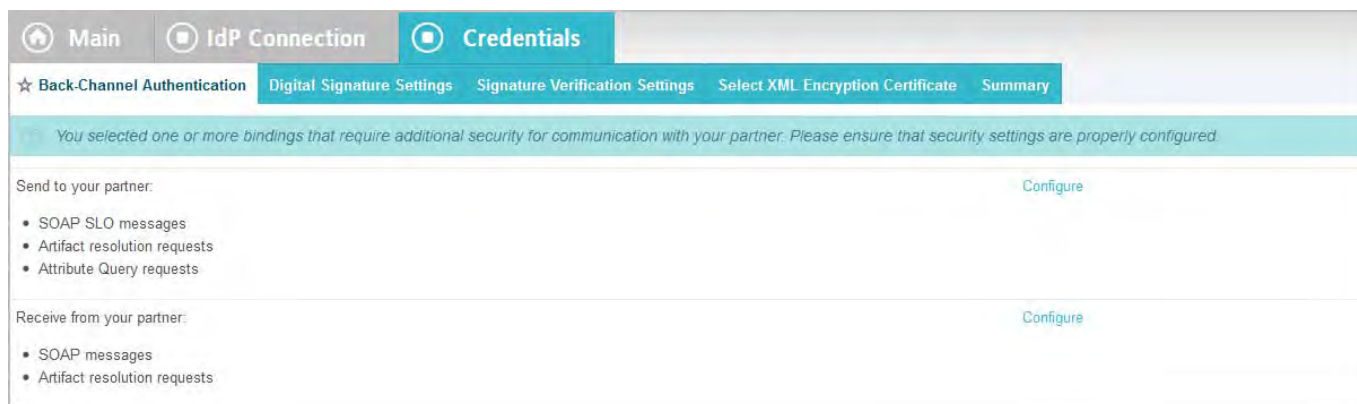
► To configure or modify credentials, click **Configure Credentials**.

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 387).



Note: A yellow triangle next to a listing indicates that you have not completely configured back-channel authentication requirements.

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.
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 225). You need to export the certificate (only) and send it 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. Optionally, you may also restrict the issuer to a specific Trusted CA to mitigate potential man-in-the-middle attacks as well as to provide a means to isolate certificates used by different connections. The CA's certificate must be imported into the PingFederate Trusted CA store (see [“Trusted Certificate Authorities”](#) on page 222).

- 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 387) 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 427).

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 423 and [“Defining Security Policy”](#) on page 427).

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 227).
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 is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection if you and your connection partner have agreed to use a stronger algorithm.

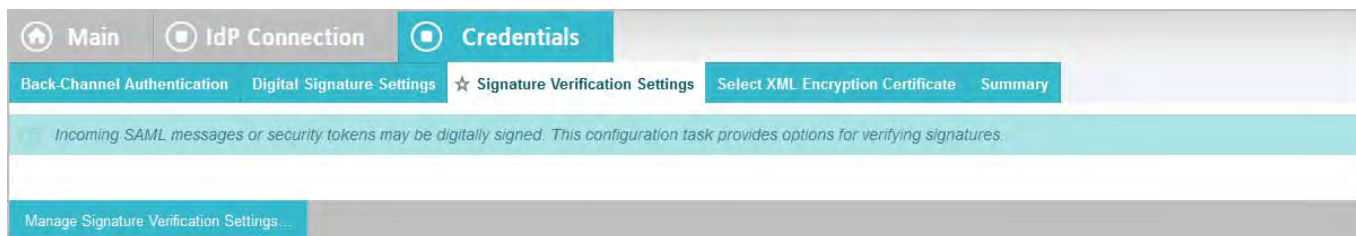
Signature Verification Settings

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 29). 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 222).

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.



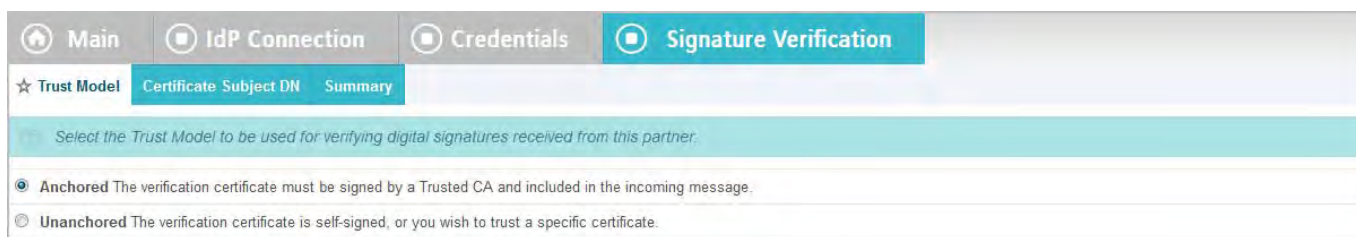
► To continue, click **Manage Signature Verification Settings**.

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 29).



- 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 461).

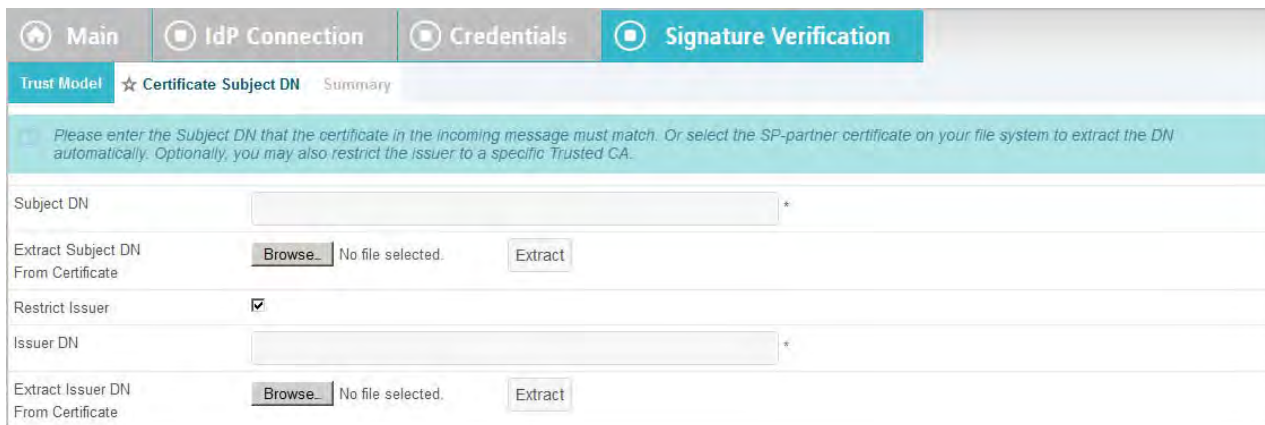
To reach this screen for editing:

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 29). PingFederate verifies that the Issuer DN (if specified) matches that of one of the issuers in the chain, the Issuer CA is trusted and the embedded certificate's Subject DN matches the one specified on this screen. If so, PingFederate uses that certificate to verify the message signature.



To complete the configuration:

1. Enter the Subject DN or extract it from your IdP partner's certificate if the certificate is stored on an accessible file system.



Tip: To extract the Subject DN from a certificate, browse to select your IdP partner's public certificate and click **Extract**.

2. (Optional) Select the Restrict Issuer checkbox. Enter the Issuer DN or extract it from a certificate if it is stored on an accessible file system.



Important: We recommend enabling this option to mitigate potential man-in-the-middle attacks as well as to provide a means to isolate certificates used by different 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 **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 **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 29).

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 **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 424.

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 427).

Main IdP Connection Credentials

Back-Channel Authentication Digital Signature Settings Signature Verification Settings ☆ Select XML Encryption Certificate Summary

Please select the partner certificate to use when encrypting message content as well as the preferred block encryption and key transport algorithms. Only RSA keys can be used for XML encryption.

Block Encryption Algorithm

AES-128

AES-256 (help)

Triple DES

Key Transport Algorithm

RSA-v1.5 (due to security risks associated with the algorithm, the use of RSA-v1.5 is strongly discouraged and not enabled for new connections)

RSA-OAEP

01:23:DD:AE:AE:11 (cn=encryption) *

Manage Certificates...

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 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 387) or you have chosen not to encrypt the assertion or the SAML_SUBJECT (see “[Configuring XML Encryption Policy \(for SAML 2.0\)](#)” on page 424).

To identify the encryption certificate:

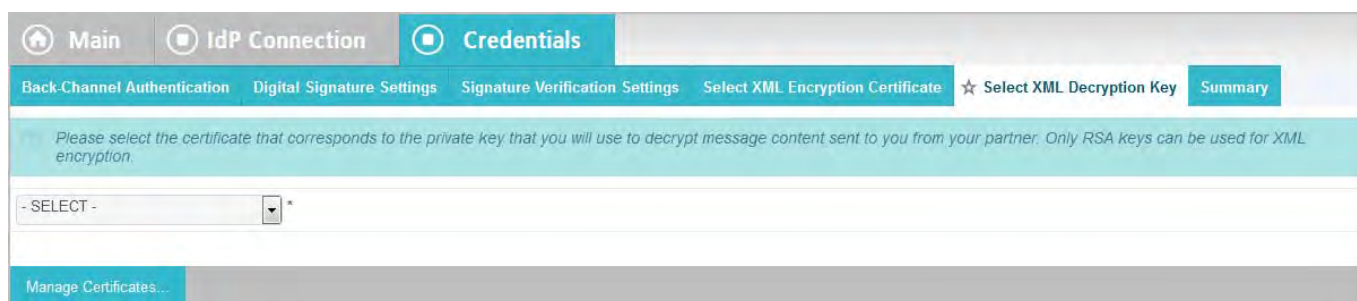
1. (Optional) Change the default settings under Block Encryption Algorithm.
Due to import control restrictions, the standard JRE distribution supports strong but not unlimited encryption. To use the strongest AES encryption, when permissible, download and install the appropriate version of “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” from the [Oracle download Web site](http://www.oracle.com/technetwork/java/javase/downloads/index.html) (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/)” (<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 424).



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 424).

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 227).

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 563.

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 32). 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 111).

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 32); there is no need to exchange certificates with your partners.

The screenshot shows the 'IdP Auto-Connect' configuration page. Under the 'Signing Certificate' tab, there is a text prompt: 'Select a key/certificate to use for signing SAML messages to Auto-Connect™ partners.' Below this, the 'Signing Certificate' dropdown is set to '01:49:AB:18:F8:74 (cn=Ping) *' and the 'Signing Algorithm' dropdown is set to 'RSA SHA256'. A 'Manage Certificates...' button is located at the bottom left of the configuration area.

You can use the same certificate used for signing metadata (see [“Configuring Auto-Connect Metadata Signing”](#) on page 120). If you use a different certificate, ensure that it meets Auto-Connect validation requirements (see [“Auto-Connect Security Model”](#) on page 34).

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 227).

2. (Optional) Select the Signing Algorithm from the drop-down list.

The default selection is RSA SHA256 or ECDSA SHA256, depending on the Key Algorithm value of the chosen Signing Certificate. Make a different selection 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.

User-Session Configuration	
Identity Mapping	Account Mapping
Attribute Contract	SAML_SUBJECT, email, fname, lname
Adapter Instances	1

- ▶ Click **Configure User-Session Creation** to continue.

For configuration information, refer to sections under “[User-Session Creation](#)” on page 390.



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 392).

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 121).

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 567), 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 32.) 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 111). 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 470).

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 114).

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 [“Token Exchange Mapping”](#) on page 155).

The screenshot shows the 'Server Settings' section with 'WS-Trust STS Settings' selected. The 'Summary' tab is active, displaying a table with 'Authentication Methods' set to 'None'. A 'Configure WS-Trust STS Authentication' button is visible at the bottom.

► 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.)

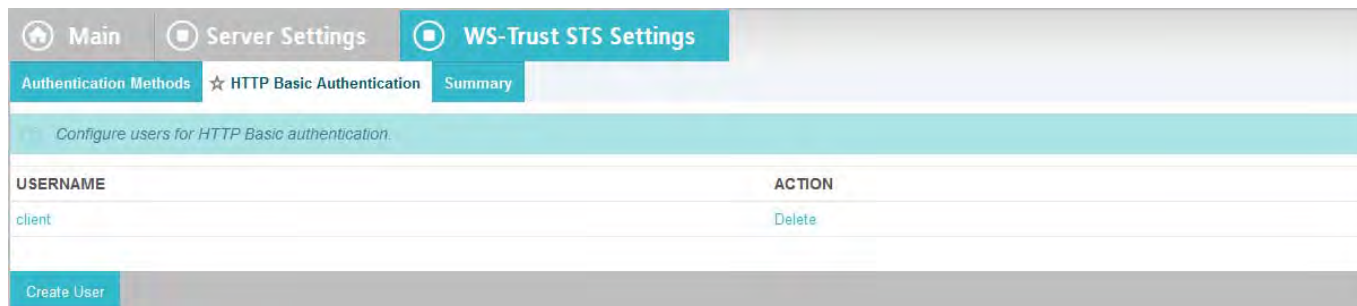


Important: If you choose mutual SSL/TLS authentication, you must configure a secondary PingFederate SSL port (see the property `pf.secondary.https.port` in the table under [“Changing Configuration Parameters”](#) on page 80)

The screenshot shows the 'Authentication Methods' tab selected under 'WS-Trust STS Settings'. It displays two checkboxes: 'Require HTTP Basic Authentication' (checked) and 'Require Mutual SSL Authentication' (unchecked). A note above the checkboxes states: 'Specify the authentication mechanism(s) that can be used to access WS-Trust STS endpoints. If both HTTP Basic Authentication and Mutual SSL Authentication are selected, then the client must provide credentials for both mechanisms.'

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 473).)
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:

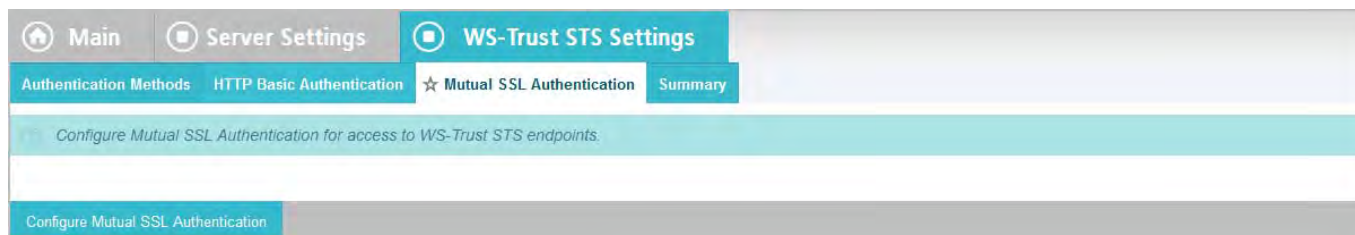
1. Click **Delete** under Action for the Username.
2. Click **Done** (or **Next** for new configuration).
3. Click **Save** when you reach the WS-Trust STS Settings screen.

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 80)



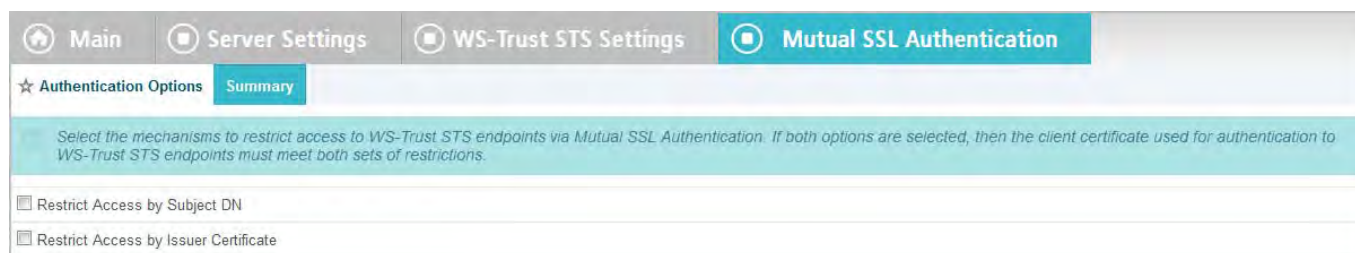
► To continue, click **Configure Mutual SSL Authentication**.

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.

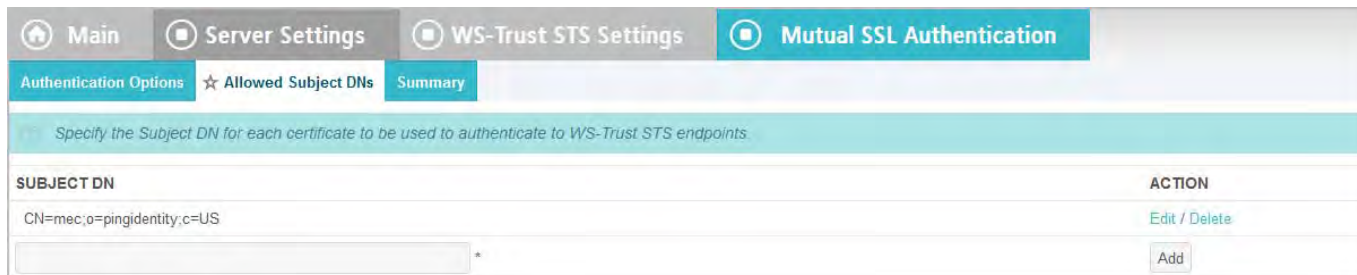


► To continue, select one or both methods and click **Next**.

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 474.

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.



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:

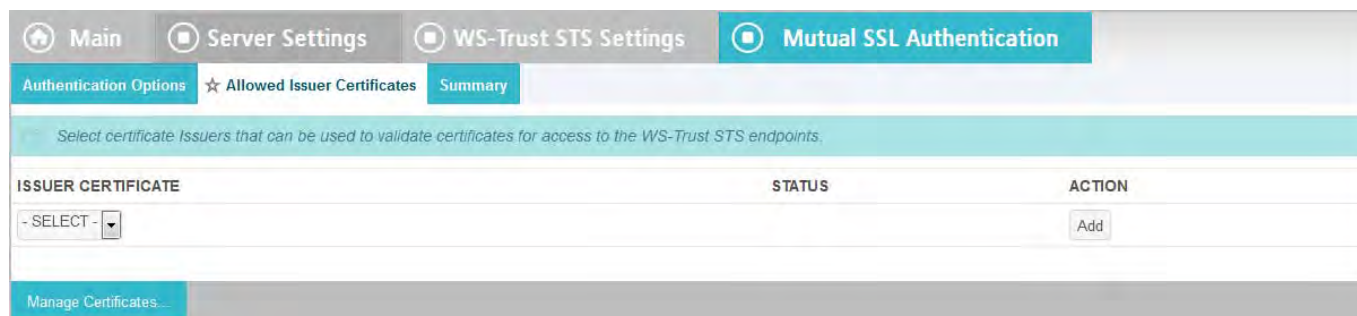
1. Click **Edit** under Action for the Subject DN.
2. Make changes and click **Update**.
3. Click **Done**.
4. If you are finished with a new or existing configuration, continue clicking **Done** until you reach the WS-Trust STS Settings screen and then click **Save**.

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.



On this screen you can add or remove 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.

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 “[Token Exchange Mapping](#)” on page 155).



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 265.)

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 469).

INSTANCE NAME	INSTANCE ID	TYPE	PARENT NAME	ACTION
OAuth Bearer AT Token Processor	OauthBearerATTP	OAuth Bearer Access Token Token Processor		Delete
SAML 1.1	saml11	SAML 1.1 Token Processor		Delete
SAML 2.0	saml20	SAML 2.0 Token Processor		None Available - In Use
SAML 2.0 Token Processor Child	saml20child	SAML 2.0 Token Processor	SAML 2.0	Delete
Username Token	UsernameTokenProcessor	Username Token Processor		None Available - In Use
Username Token Child	UsernameTokenChild	Username Token Processor	Username Token	Delete

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 18). 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 479
- [“Configuring a JSON Web Token \(JWT\) Processor Instance”](#) on page 479
- [“Configuring a Username Token Processor Instance”](#) on page 480

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.

Home Main Manage Token Processor Instances **Create Token Processor Instance**

Type ☆ Instance Configuration Extended Contract Token Attributes Summary

Complete the configuration necessary for this token processor in your environment.

SAML 2.0 Token Processor

VALID CERTIFICATE ISSUER DNS (If Issuer DNs are specified here (optional), then only those issuers are considered valid for verifying incoming digital signatures. Otherwise, all trusted Certificate Authorities (CAs) are used to verify signatures.)

ISSUER DN (List of Acceptable Root CA Issuers) Action

Add a new row to 'Valid Certificate Issuer DNs'

VALID CERTIFICATE SUBJECT DNS (If Subject DNs are specified here (optional), then only matching certificates are considered valid for verifying incoming digital signatures. Otherwise, all trusted CAs—or those listed above—are used to verify signatures.)

SUBJECT DN (List of Acceptable Certificates) Action

Add a new row to 'Valid Certificate Subject DNs'

FIELD NAME	FIELD VALUE	DESCRIPTION
AUDIENCE	<input type="text"/>	Provide the Uniform Resource Identifier (URI) that uniquely identifies your federation gateway for this SAML protocol. The incoming SAML <audience> value must match this URI.



Note: If this is a child instance, select the override checkbox to modify the 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 114).
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 175).

FIELD NAME	FIELD VALUE	DESCRIPTION
ACCESS TOKEN MANAGER	-- Select One --	The Access Token Management plugin instance that will be used to validate the OAuth Bearer Access Token.
SCOPE VALUE AS SINGLE STRING	<input type="checkbox"/>	If selected, the Scope value will be made available as a single space delimited set of string values. Alternatively, Scope values will be made available as a multivalue attribute.



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

Field Descriptions

Field	Description
Access Token Manager	Select the Access Token Management plug-in instance that will be used to validate the OAuth Bearer Access Token.
Scope Value as Single String (optional)	If selected, the Scope value will be made available as a single space delimited set of string values. Alternatively, Scope values will be made available as a multivalue attribute.

To configure the token-processor instance for OAuth Bearer token validation:

1. Select an Access Token Manager.
2. (Optional) Select the Scope Value as Single String checkbox.
3. Click **Next**.

Configuring a JSON Web Token (JWT) Processor Instance

The PingFederate STS provides validation for JSON web tokens.

🏠 Main
⊞ Manage Token Processor Instances
⊞ Create Token Processor Instance

Type
☆ Instance Configuration
Extended Contract
Token Attributes
Summary

Complete the configuration necessary for this token processor in your environment.

JWT Token Processor 1.0

FIELD NAME	FIELD VALUE	DESCRIPTION
JWKS ENDPOINT URI	<input type="text"/>	A set of JSON Web Keys (JWKS) are downloaded from this endpoint and used for JWT signature verification.
ISSUER	<input type="text"/>	* A unique identifier for the issuer of the JWT.
EXPIRY TOLERANCE	<input type="text" value="0"/>	The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.



Note: If this is a child instance, select the override checkbox to modify the configuration.

Field Descriptions

Field	Description
JWKS Endpoint URI	The URI of the JWKS endpoint. A set of JSON Web Keys (JWK) are downloaded from this endpoint and used for JWT signature verification.
Issuer	(Required) A unique identifier for the issuer of the JWT.
Expiry Tolerance	The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.

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).

Home Main
Manage Token Processor Instances
Create Token Processor Instance

Type
★ Instance Configuration
Extended Contract
Token Attributes
Summary

Complete the configuration necessary for this token processor in your environment.

A token processor that validates username tokens against a set of Password Credential Validator instances. The first Credential Validator that successfully validates a token will pass its attributes to this token processor and can be accessed by extending the attribute contract.

PASSWORD CREDENTIAL VALIDATORS (A list of Credential Validator instances to authenticate Username Tokens. A token is processed in the same order as the selected Credential Validators)

CREDENTIAL VALIDATOR	Action
(A configured Password Credential Validator instance)	
Add a new row to 'Password Credential Validators'	

FIELD NAME	FIELD VALUE	DESCRIPTION
This plugin type has no individual configurable fields.		

[Manage Password Credential Validators...](#)



Note: If this is a child instance, select the override checkbox to modify the configuration.

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 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](#).

 **Note:** This screen shows a different list of attributes for the Core Contract, depending on the Token Processor selected.

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 182).

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 25).

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 613).

Editing and Saving Processor Instances

From the Summary screen, you can reach processor settings for editing.

To edit the configuration:

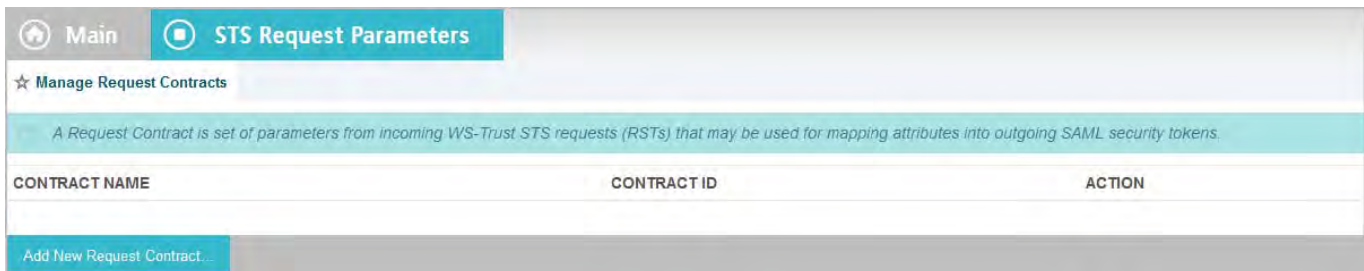
1. Click the heading above the information you want to change.
2. Make your changes.
3. Click **Done** on the configuration page and **Save** on the Manage Token Processors screen.

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 488).



- ▶ 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 469).
- ▶ 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 483).

☆ Create Request Contract

Specify one or more parameters that will be included in RSTs applicable to a connection partner (or partners). You can make Request Contracts available for token-attribute mapping during partner-connection configuration.

Contract Name *

Contract ID *

Parameters to be provided in the request

PARAMETER NAME	ACTION
<input type="text"/> *	<input type="button" value="Add"/>

Field Descriptions

Field	Description
Contract Name	A descriptive name for the Contract—for example, a Web Service Client or Provider.
Contract ID	An internal identifier—must be alphanumeric with no spaces.
Parameters to be provided in the request	A list of request parameters for this Contract (see instructions below).

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.

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 273).



Note: Before this option can be selected, the WS-Trust protocol must be enabled in Server Settings (see “Server Settings” on page 469)

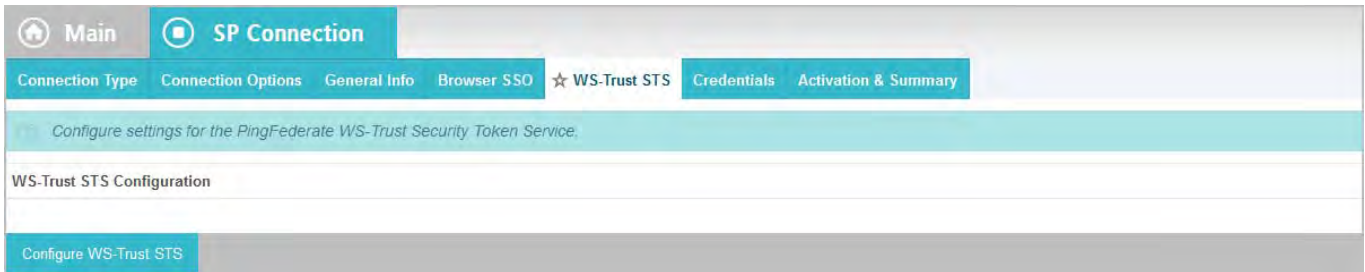
2. Select a Default Token Type.

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).

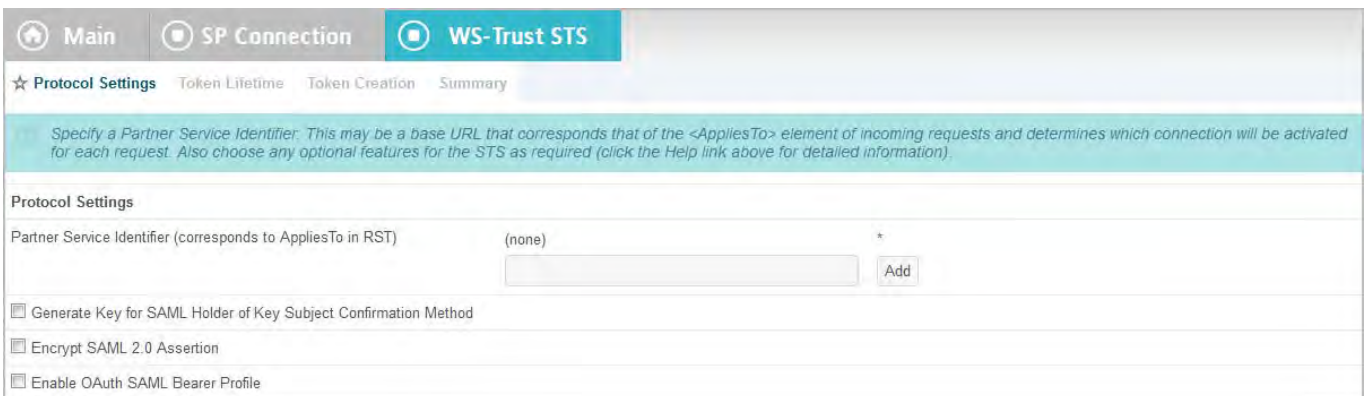
When the option is enabled, the configuration starts on the WS-Trust STS screen.



- 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](http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0.pdf)” (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 484).

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 330.)

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.

Main		SP Connection		WS-Trust STS	
Protocol Settings		★ Token Lifetime		Token Creation Summary	
When a token is issued to the SP, there is a timeframe of validity before and after issuance. Please specify these parameters below.					
Minutes Before	<input type="text" value="5"/>	*			
Minutes After	<input type="text" value="30"/>	*			

Field Descriptions

Field	Description
Minutes Before	The amount of time before the token was issued during which it is to be considered valid.
Minutes After	The amount of time after the token was issued during which it is to be considered valid.

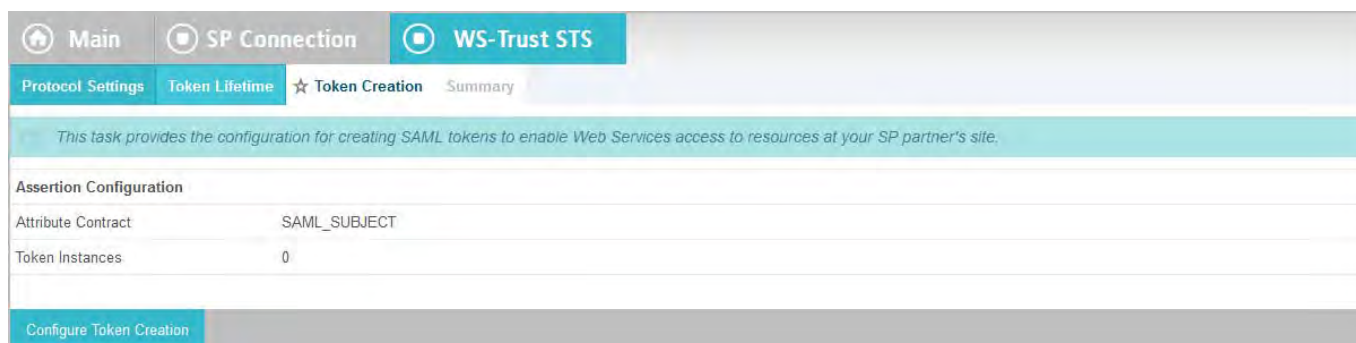
To change the default times:

- ▶ (Optional) Edit the desired setting(s) and click **Next** or **Save**.

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 503). As with Browser SSO, the mapping can be augmented using local data stores, variable or constant text, or expressions.

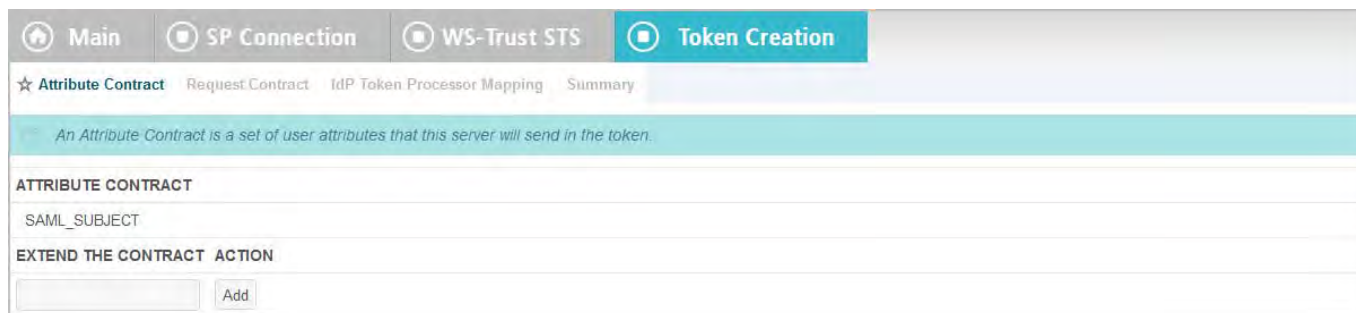
Details of this configuration are handled under the Token Creation task.



► 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 21). 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 503).

For information about the NameID elements and applicable URI values, locate the SAML 2.0 specification at oasis-open.org/specs.



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 503).

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 484).

Change the default Namespace selection if you and your SP partner have agreed to a specific namespace (see [“STS Namespaces”](#) on page 22).



Note: If needed, an administrator can customize namespace alternatives via the `custom-name-formats.xml` configuration file located in this directory:

```
<pf_install>/pingfederate/server/default/data/  
config-store
```

3. Click **Add**.

To modify an attribute name or namespace:

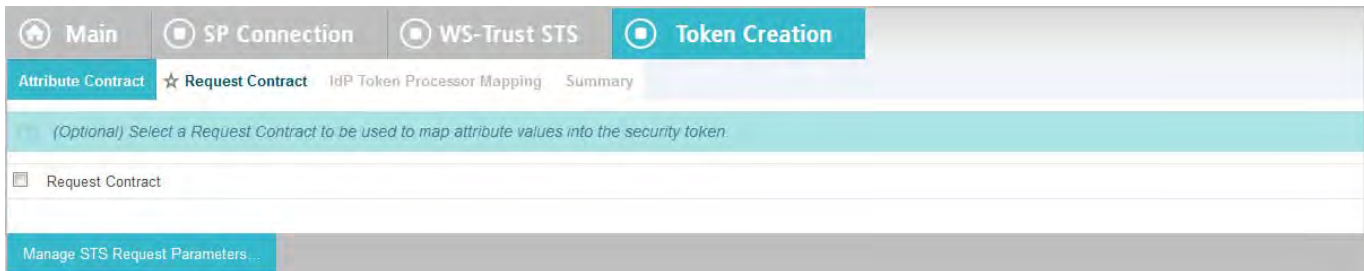
1. Click **Edit** under Action for the attribute.
2. Make the change and click **Update**.

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 483).



- ▶ 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 503).

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.



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 518).

★ **Token Processor Instance** Attribute Retrieval Attribute Contract Fulfillment Issuance Criteria Summary

Select an IdP token processor instance that may be used to authenticate users for this partner. Attributes returned by the token processor instance you choose (the Token Processor Contract) may be used to fulfill the Attribute Contract with your partner.

TOKEN PROCESSOR INSTANCE - SELECT -

TOKEN PROCESSOR CONTRACT

Override Instance Settings

Manage Token Processor Instances

- 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 490).

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

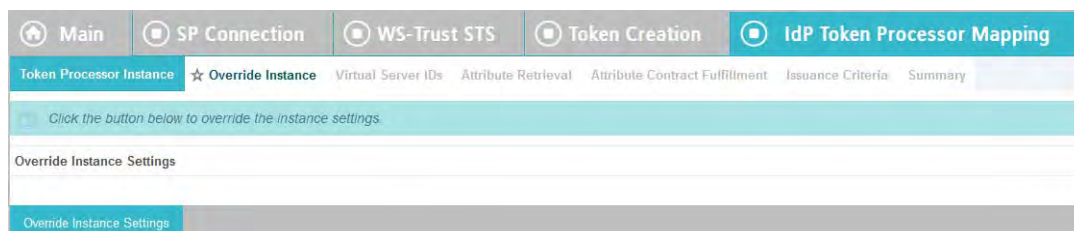
Overrides at the connection level simplify token management by allowing you to create a small set of token instances that define the base configuration requirements and then overriding settings at the connection level as needed.

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 18 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:

Override Screen	Manual Section
Instance Configuration	See “Configuring Token Processors” on page 475.
Extended Contract	See “Extending a Processor Contract” on page 481.
Token Attributes	See “Setting Attribute Masking” on page 482.

- 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 39), 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 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 38.

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 475).

The values supplied by the token processor are shown under Token Processor Contract on the Attribute Retrieval 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 **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 493).
 - ▶ 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 503).



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 487). 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 503.)

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 487). 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.

DESCRIPTION	TYPE	ACTION
Add Attribute Source...		

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 492).

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 122). 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 487).

The screenshot shows a web interface with a navigation bar at the top containing tabs: Main, SP Connection, WS-Trust STS, Token Creation, and IdP Token Processor Mapping. Below the navigation bar, there is a breadcrumb trail: ☆ Data Store > Summary. A teal banner contains the text: "This server uses local data stores to retrieve supplemental attributes returned by PingFederate from the User Info Endpoint and within the ID Token. Specify an Attribute Source name that will distinguish this user lookup for the selected data store." Below the banner are four form fields: "Attribute Source Id" (text input), "Attribute Source Description" (text input), "Active Data Store" (dropdown menu with "- SELECT -" selected), and "Data Store Type" (text input with "None" selected). At the bottom left, there is a button labeled "Manage Data Stores".

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 492).
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 503).

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 122).

Setting Up the Attribute Source

See the following sections in this manual, depending on the type of data store:

Data Store Type	Related Manual Section
JDBC	<ul style="list-style-type: none"> • “Selecting an STS JDBC Database Table and Columns” on page 495 • “Configuring an STS Database Filter” on page 497
LDAP	<ul style="list-style-type: none"> • “Configuring an LDAP Search” on page 499 • “Configuring an LDAP Filter for STS” on page 501
Custom	<ul style="list-style-type: none"> • “Configuring STS Custom Source Filters” on page 503 • “Selecting Custom STS Source Fields” on page 503

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 487). 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:

```
jdbc:mysql://myhost.mydomain.com:3306/
pf?sessionVariables=sql_mode=ANSI_QUOTES
```

For more information, see:

dev.mysql.com/doc/refman/5.0/en/identifiers.html

and

dev.mysql.com/doc/refman/5.1/en/option-files.html

The screenshot shows the WS-Trust STS configuration page. The navigation tabs include Main, SP Connection, WS-Trust STS (selected), Token Creation, and IdP Token Processor Mapping. The sub-tabs under WS-Trust STS are Data Store, Database Table and Columns (selected), Database Filter, and Summary. A message states: "Please select the table and columns you want to query. This information, along with the attributes supplied in the contract, will be used to fulfill the contract." The form fields are: Schema (dropdown set to 'dbo'), Table (dropdown set to 'USERS'), Columns to return from SELECT (dropdown set to 'EMAIL'), and an 'Add Attribute' button. There is also a 'Refresh' button and a 'View Attribute Contract' link at the bottom.

Field Descriptions

Field	Description
Schema	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.
Table	The name of the table contained in the database. Use the drop-down to change the table.
Columns to return from SELECT	Displays selected table columns. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.

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.



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 487). Then determine what information is coming in from the token processor (see [“Retrieving Attributes”](#) on page 492). Information not contained in the token-processor contract may be pulled from the data store look-up query.

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 495).

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 475).



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 24.

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.

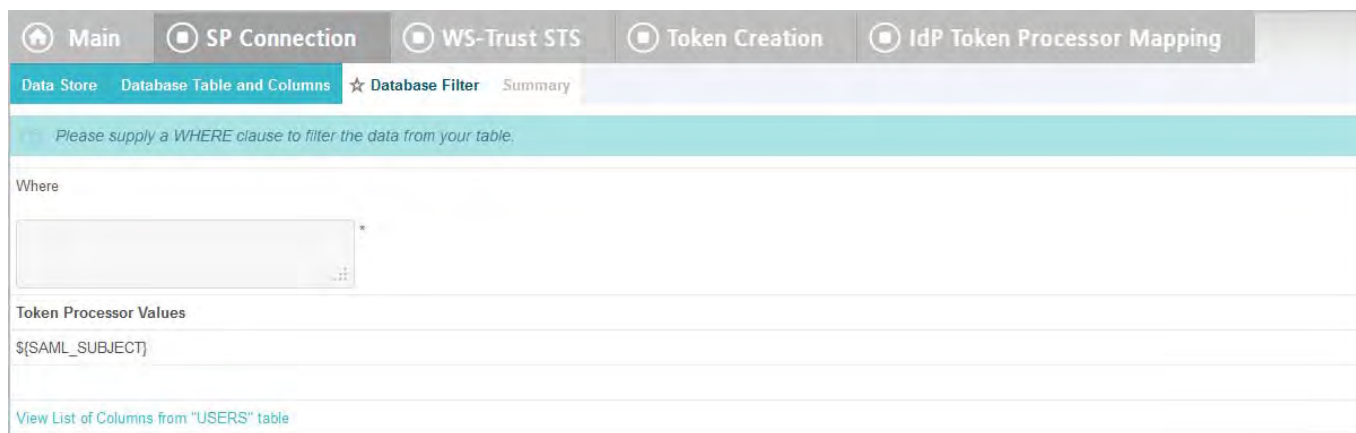
EXAMPLE:

`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

Field	Description
Where	WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: WHERE email='clive@company.com'.

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.

- Click **Database Filter** under the IdP Token Processor Mapping tab.

To construct the WHERE clause:

- Enter the statement in the space provided, following the guidelines and example above.

The initial WHERE is optional.

- 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 503).

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.

Field Descriptions

Field	Description
Base DN	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.
Search Scope	Determines the node depth of the query. Select Subtree, One level or Object.
Root Object Class	The class containing the attributes you want.
Attributes to return from search	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.

To reach this screen for editing:

- Click the connection name on the Main Menu.
Click **Manage All SP**, if needed, to see a full list of connections.
- 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.



Note: When connecting to Microsoft Active Directory, if you choose the memberOf attribute, an optional checkbox, Nested Groups, appears on the right. Select this checkbox if you want PingFederate to query for groups the end users belong to directly as well as indirectly through nested group membership (if any) under the Base DN.

For example, suppose you have three groups under the Base DN, namely Canada, Washington and Seattle. Seattle is a member of Washington. Ana Smith is an end user and a member of Seattle. If the Nested Groups checkbox is selected, when PingFederate queries for Ana's memberOf attribute values, the expected results are Seattle and Washington. (When the Nested Groups checkbox is not selected (the default), the expected result is Seattle.)

For Oracle Directory Server, choose isMemberOf under Attribute for nested group membership. For more information, see [documentation about isMemberOf from Oracle](https://docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm) (docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm).

5. Repeat the last step for other attributes as needed.



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, Hex or SID) 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.

Claims-based authentication with Microsoft Outlook Web App and Exchange admin center (EAC) is another example. It requires tokenGroups (another binary attribute in Active Directory) to be SID-encoded.



Note: Attributes are specified as binary when configuring an LDAP connection (see [“Specifying LDAP Binary Attributes”](#) on page 132).

Main		SP Connection		WS-Trust STS		Token Creation		IdP Token Processor Mapping	
Data Store	LDAP Directory Search	★ LDAP Binary Attribute Encoding Types		LDAP Filter	Summary				
Please specify the encoding type for the selected binary directory attributes.									
ATTRIBUTE NAME					ATTRIBUTE ENCODING TYPE				
objectGUID					Base64				

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:

$$attribute=\${value}$$

The left-side variable is an attribute you selected earlier (see [“Configuring an LDAP Search”](#) on page 499).

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 475).

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.

The screenshot shows a web-based configuration interface for WS-Trust STS. At the top, there are navigation tabs: Main, SP Connection, WS-Trust STS, Token Creation, and IdP Token Processor Mapping. Below these, there are sub-tabs: Data Store, LDAP Directory Search, LDAP Filter (selected), and Summary. A teal banner at the top of the main content area contains the text: "Please enter a Filter for extracting data from your directory." Below this is a text input field labeled "Filter". Underneath the input field is a section titled "Token Processor Values" containing the text "\${SAML_SUBJECT}". At the bottom left of the form, there is a link that says "View List of Available LDAP Attributes".

Field Descriptions

Field	Description
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 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.

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: If you used an anonymous binding to create this LDAP connection, your access might be restricted (see "[Configuring an LDAP Connection](#)" on page 127).

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.

Main				SP Connection		WS-Trust STS		Token Creation		IdP Token Processor Mapping	
Token Processor Instance		Attribute Retrieval		★ Attribute Contract Fulfillment		Issuance Criteria		Summary			
Fulfill your Attribute Contract with values from the incoming token, data stores, or dynamic text values.											
ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS								
SAML_SUBJECT	- SELECT -		None available								

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 493).

Values are returned from your attribute source (if you are using data store—see [“Retrieving Attributes”](#) on page 492). 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 499, [“Selecting an STS JDBC Database Table and Columns”](#) on page 495, or [“Configuring STS Custom Source Filters”](#) on page 503).

- 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 618). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 613. For syntax and examples, see sections under “Constructing Expressions” on page 614.)



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 471).

- 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 488).

- 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 613). 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.oauth2.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 493) 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 25). This token authorization can be used to restrict who can access identity-enabled Web services.

SOURCE	ATTRIBUTE NAME	CONDITION	VALUE	ERROR RESULT	ACTION
- SELECT - *	- SELECT - *	- SELECT - *	*	*	Add

Show Advanced Criteria

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.



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 493).

- Mapped Attributes – Select to access the required attributes.
 - 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 471).

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.

For more information on SOAP, see the World Wide Web Consortium’s [Simple Object Access Protocol](#) (www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507).



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 613).



Important: When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), 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 616).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618. 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.



Note: The Error Handling screen is presented only if a Request Contract is used for this configuration (see [“Selecting a Request Contract”](#) on page 488).

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.

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 [“Token Exchange Mapping”](#) on page 155).



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 374.)

For example:

```
https://<pf_host>:<pf_port>/sp/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 469).

★ Manage Token Generators

PingFederate uses token generators to issue tokens that can be used by Web Services at your site to authenticate service requests. Create "instances" of token generators here to use within IdP connections for mapping attributes from incoming SAML tokens to attributes required by local applications.

INSTANCE NAME	INSTANCE ID	TYPE	PARENT NAME	ACTION
SAML 1	saml1	SAML 1.1 Token Generator		Delete
SAML 2	saml2	SAML 2.0 Token Generator		None Available - In Use
Saml 2 child	saml2child	SAML 2.0 Token Generator	SAML 2	Delete

Create New Instance...

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 18). 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.

[Main](#) | [Manage Token Generator Instances](#) | **Create Token Generator Instance**

Type: [★ Instance Configuration](#) | [Extended Contract](#) | [Summary](#)

Complete the configuration necessary to set the appropriate security token for access to Web Services in your environment.

SAML 2.0 Token Generator

FIELD NAME	FIELD VALUE	DESCRIPTION
MINUTES BEFORE	<input type="text"/> *	The earliest time for which the generated SAML token will be valid.
MINUTES AFTER	<input type="text"/> *	The latest time for which the generated SAML token will be valid.
ISSUER	<input type="text"/> *	A unique identifier for the published SAML authority generating the token.
SIGNING CERTIFICATE	<input type="text" value="-- Select One --"/> *	Select the signing certificate that will be used to sign the generated assertion.
INCLUDE CERTIFICATE IN KEYINFO	<input type="checkbox"/>	Include the certificate in the signature <KeyInfo> element.
INCLUDE RAW KEY IN KEYVALUE	<input type="checkbox"/>	Include the raw key in the signature <KeyValue> element.
AUDIENCE	<input type="text"/> *	A URI reference that identifies the intended audience of the generated assertion.
CONFIRMATION METHOD	<input type="text" value="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches"/> ▼	The method for verifying the SAML subject in the token.
ENCRYPTION CERTIFICATE	<input type="text" value="-- Select One --"/> ▼	The encryption certificate to be used for the holder-of-key Confirmation Method.

[Manage Signing Certificates...](#) | [Manage Encryption Certificates...](#)



Note: If this is a child instance, select the override checkbox to modify the configuration.

Field Instructions

Field	Instruction
Minutes Before	Enter a numerical value. This element in a SAML token allows for any server clock variability.
Minutes After	Enter a numerical value. This element in a SAML token allows for any server clock variability.
Issuer	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 114).
Signing Certificate	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.
Include Certificate in KeyInfo	If selected, the entire public certificate is included with the assertion. Otherwise, a short hash reference to the certificate is sent instead.
Include Raw Key in KeyValue	If selected, the raw key is included in the <KeyInfo> element as well.

Field	Instruction
Audience	This is a unique identifier for the target Web service, used for the <audience> element of the generated SAML token.
Confirmation Method	(Optional) Choose from among available methods: <ul style="list-style-type: none"> • ...cm:sender-vouches (default) • ...cm:bearer • ...cm:holder-of-key For more information, see the WSS SAML Token Profile .
Encryption Certificate	The WSP's public certificate for encryption, required <i>only</i> if holder-of-key is selected as the Confirmation Method. If the certificate is not yet part of the PingFederate store, click Manage Encryption Certificates to import it.

Extending a Generator Contract

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.



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**.

Editing and Saving Generator Instances

From the Summary screen, you can reach token-generator 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 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.

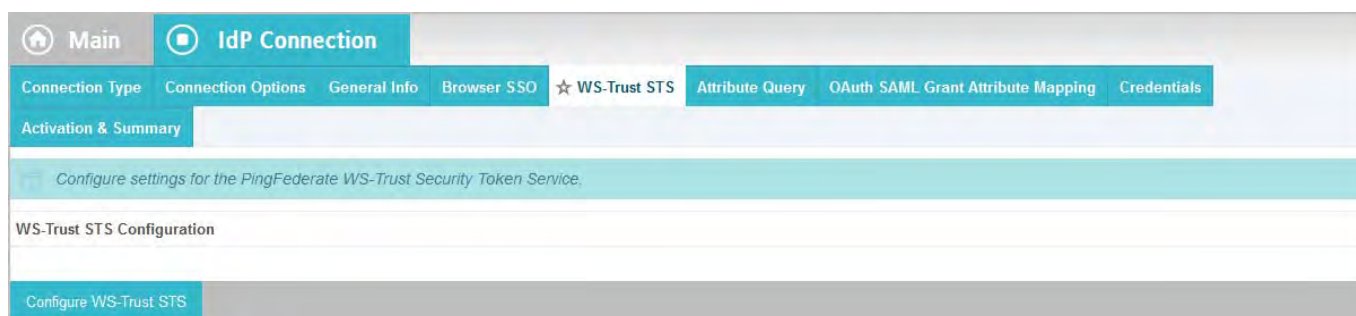
To enable STS for a new connection, or to add the capability to an existing connection:

- ▶ Select the WS-Trust STS option on the Connection Type screen (see “[Choosing an IdP Connection Type](#)” on page 381).



Note: Before this option can be selected, the WS-Trust protocol must be enabled in Server Settings (see “[Server Settings](#)” on page 469)

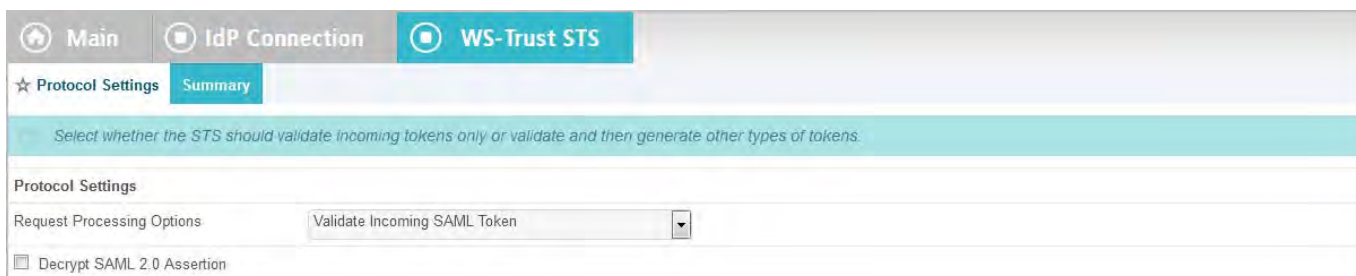
When the option is enabled, the configuration starts on the WS-Trust STS screen.



- ▶ 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 462).

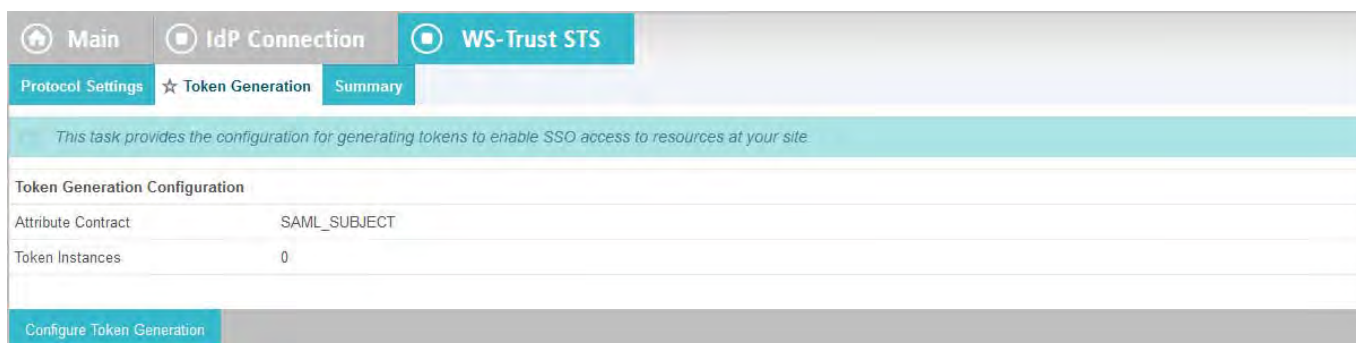
- ▶ If you choose not to generate new tokens, then no further settings are needed for this task—click **Next** and refer to “[Using the Token Generation Summary Screen](#)” on page 532 for instructions on saving this configuration.

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 334).

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 527). 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.

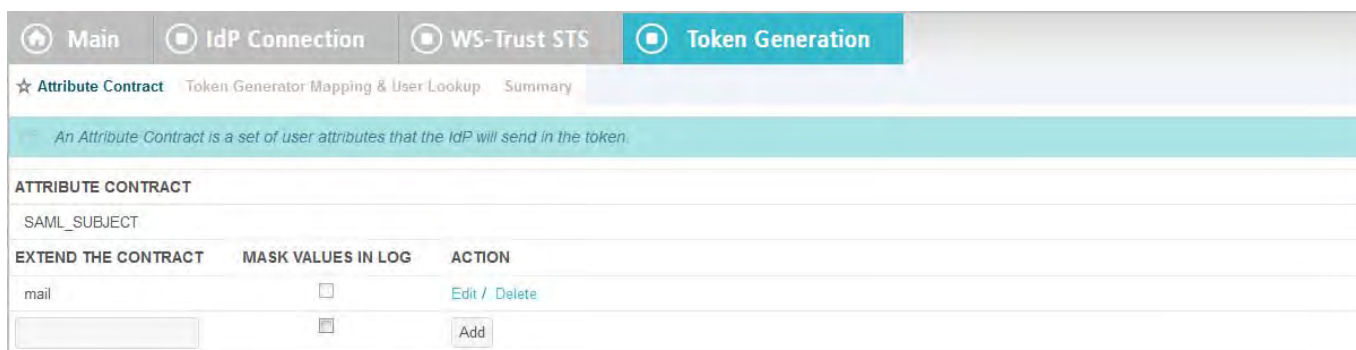


► To continue, click **Configure Token Creation**.

Specifying an Attribute Contract

An attribute contract is the set of user attributes expected in incoming security tokens (see “[Attribute Contracts](#)” on page 21). 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 25).



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**.

- 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 513).

- Click **Configure Token Generation**
- Click **Attribute Contract** on the Summary screen.

To add an attribute:

- Enter the attribute name in the text box.
Attribute names are case-sensitive and must correspond to the attribute names expected by the requester.
- (Optional) Select the checkbox under Mask Values in Log.
- Click **Add**.

To modify an attribute name:

- Click **Edit** under Action for the attribute.
- 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 509.

Token Generation Configuration	VIRTUAL SERVER IDS	ACTION
Saml11Token	SPa SPb	Set as Default Delete
Saml20Token		Default Delete

Map New Token Generator Instance...

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 513).
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 492).

- ▶ 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 516).

To create or change a Token Generator Instance, as needed, click **Manage Token Generator Instances**.

Overriding Token Generator Instances

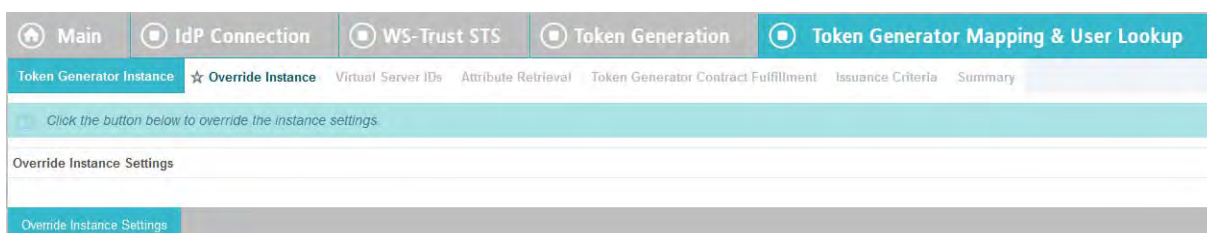
Overrides at the connection level simplify token management by allowing you to create a small set of token instances that define the base configuration requirements and then overriding settings at the connection level as needed.

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 18 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.

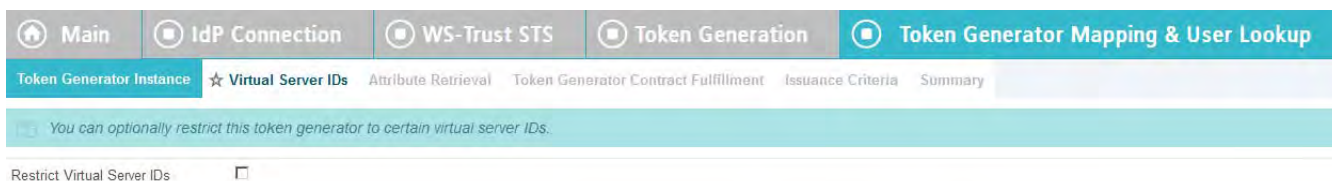
For information about the override token-setting screens, depending on the type of setting, use the following table:

Override Screen	Manual Section
Instance Configuration	See “Configuring a Token Generator Instance” on page 510.
Extended Contract	See “Extending a Generator Contract” on page 512.

- ▶ 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 39), 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 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 513).
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 38.

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.

Main	IdP Connection	WS-Trust STS	Token Generation
Token Generator Instance	★ Attribute Retrieval	Token Generator Contract Fulfillment	Issuance Criteria Summary
<p>You can fulfill the Token Generator Contract by using only the attributes from the incoming SAML token or by using these attributes to look up additional information from a local data store.</p>			
ATTRIBUTE CONTRACT			
SAML_SUBJECT			
mail			
<input type="radio"/> Use the incoming token to look up additional information			
<input checked="" type="radio"/> Use only the attributes available in the incoming token			

To reach this screen for editing:

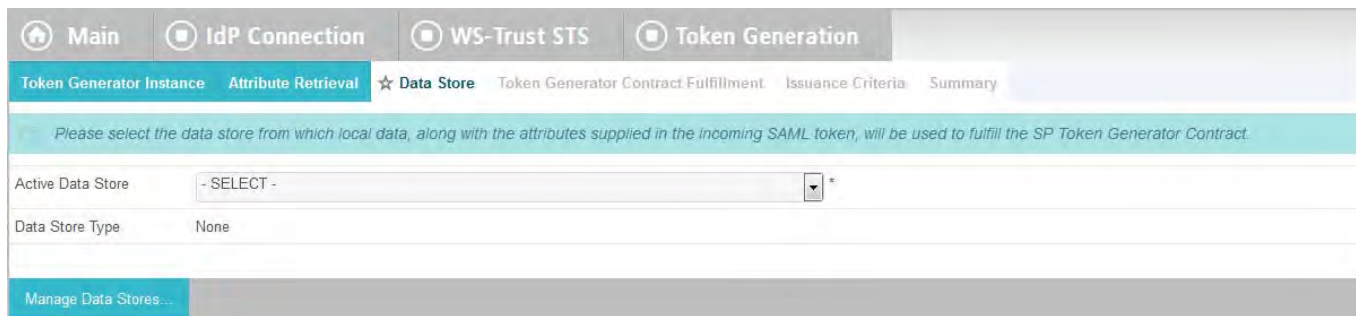
1. Click the connection name on the Main Menu.
Click **Manage All IdP**, if needed, to see a full list of connections.
2. Click **WS-Trust STS** under the IdP Connection tab.
3. Click **Configure WS-Trust STS**.
4. Click **Token Generation** under the WS-Trust STS tab.
If this step is not shown, token generation is not selected for the connection (see [“Configuring STS Protocol Settings”](#) on page 513).
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 519).
 - ▶ 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 527).



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 516). 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 527.)

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 513).
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 518).

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 122).

Configuring Attribute Lookup for WS-Trust STS

See the following sections in this manual, depending on the type of data store:

Data Store Type	Related Manual Section
JDBC	<ul style="list-style-type: none"> • “Selecting a Data Table and Columns” on page 521 • “Configuring a Database Query” on page 522
LDAP	<ul style="list-style-type: none"> • “Configuring LDAP Search Parameters” on page 524 • “Configuring a Directory Filter” on page 526
Custom	<ul style="list-style-type: none"> • “Configuring Custom Filters” on page 527 • “Selecting Custom Data Fields” on page 527

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 518). 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:

```
jdbc:mysql://myhost.mydomain.com:3306/
pf?sessionVariables=sql_mode=ANSI_QUOTES
```

For more information, see:

dev.mysql.com/doc/refman/5.0/en/identifiers.html

and

dev.mysql.com/doc/refman/5.1/en/option-files.html

Field Descriptions

Field	Description
Schema	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.
Table	The name of the table contained in the database. Use the drop-down to change the table.
Columns to return from SELECT	Displays selected table columns. Select the columns that are associated with the desired attributes you would like to return from the JDBC query.

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 513).
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 514). Then determine what information is coming in from the token processor (see [“Retrieving Additional Attributes”](#) on page 518). 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 521).

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

Field	Description
Where	WHERE clause statements conditionally select data from a table. Enter the WHERE clause statement in the space provided. For example: WHERE email='clive@company.com'.

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 513).
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 527).

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.

The screenshot shows the configuration page for LDAP Directory Search. The breadcrumb trail is: Main > IdP Connection > WS-Trust STS > Token Generation > LDAP Directory Search. The page title is "Please configure your directory search. This information, along with the attributes supplied in the incoming SAML token, will be used to fulfill the Token Generator Contract." The form includes a text input for "Base DN", a dropdown for "Search Scope" (set to "Subtree"), and a table for "Attributes to return from search". The table has columns for "ROOT OBJECT CLASS", "ATTRIBUTE", and "ACTION". One row is visible with "Subject DN" in the attribute column and an "Add Attribute" button in the action column. A "View Token Generator Contract" link is at the bottom.

Field Descriptions

Field	Description
Base DN	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.
Search Scope	Determines the node depth of the query. Select Subtree, One level or Object.

Field	Description
Root Object Class	The class containing the attributes you want.
Attributes to return from search	A list of added from the drop-down list below. Subject DN is a default attribute, which may be used as the primary user identifier.

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 513).
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.



Note: When connecting to Microsoft Active Directory, if you choose the memberOf attribute, an optional checkbox, Nested Groups, appears on the right. Select this checkbox if you want PingFederate to query for groups the end users belong to directly as well as indirectly through nested group membership (if any) under the Base DN.

For example, suppose you have three groups under the Base DN, namely Canada, Washington and Seattle. Seattle is a member of Washington. Ana Smith is an end user and a member of Seattle. If the Nested Groups checkbox is selected, when PingFederate queries for Ana’s memberOf attribute values, the expected results are Seattle and Washington. (When the Nested Groups checkbox is not selected (the default), the expected result is Seattle.)

For Oracle Directory Server, choose isMemberOf under Attribute for nested group membership. For more information, see [documentation about isMemberOf from Oracle](https://docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm) (docs.oracle.com/cd/E29127_01/doc.111170/e28967/ismemberof-5dsat.htm).

- Repeat the last step for other attributes as needed.



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 524).

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

Field	Description
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 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.

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 513).
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 127).

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.

TOKEN GENERATOR CONTRACT	SOURCE	VALUE	ACTIONS
SAML_SUBJECT	Assertion	SAML_SUBJECT	None available

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 618). (If the Expression selection is not listed, then the feature is not enabled—see “Enabling and Disabling Expressions” on page 613. For syntax and examples, see sections under “Constructing Expressions” on page 614.)



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 471).

- LDAP/JDBC/Custom
Values are returned from the selected data store (see “Retrieving Additional Attributes” on page 518). 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 524, “Selecting a Data Table and Columns” on page 521, or “Configuring Custom Filters” on page 527).
- 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 613). 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 513).
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 25). This token authorization can be used to restrict who can access identity-enabled Web services.

The screenshot shows the 'Issuance Criteria' configuration screen. At the top, there are navigation tabs: Main, IdP Connection, WS-Trust STS, and Token Generation. Below these are sub-tabs: Token Generator Instance, Attribute Retrieval, Token Generator Contract Fulfillment, Issuance Criteria (selected), and Summary. A descriptive text box states: "PingFederate can evaluate various criteria to determine whether users are authorized to access SP resources. Use this optional screen to configure the criteria for use with this token authorization." Below this is a table with the following columns: SOURCE, ATTRIBUTE NAME, CONDITION, VALUE, ERROR RESULT, and ACTION. Each of the first four columns has a dropdown menu with "- SELECT -" and an asterisk. The ACTION column has an "Add" button. At the bottom left, there is a "Show Advanced Criteria" button.

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 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 (see [“Configuring STS Attribute Sources and User Lookup”](#) on page 493).

- Mapped Attributes – Select to access the required attributes.
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 471).

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.

For more information on SOAP, see the World Wide Web Consortium’s [Simple Object Access Protocol](#) (www.w3.org/TR/2000/NOTE-SOAP-20000508/#_Toc478383507).



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 613).



Important: When you multiplex one connection for multiple environments (see [“Connecting to a Partner in One Connection”](#) on page 39), 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 616).

- Use the in-line editor box to enter the OGNL expression.
For more information about OGNL, see [“Using Attribute Mapping Expressions”](#) on page 613.
- 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 618. 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 15).

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.

The following figure shows a basic IdP-initiated SSO scenario using PingFederate with the Java Integration Kit on both sides of an identity federation.

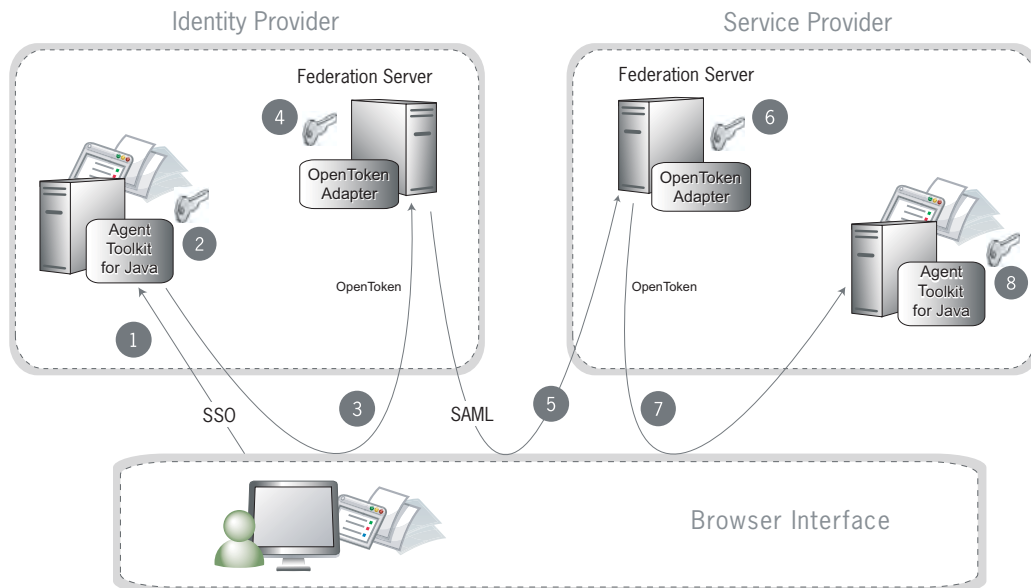


Figure 4: SP-Initiated SSO: POST/POST

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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
5. Click **Next**.

FIELD NAME	FIELD VALUE	DESCRIPTION
PASSWORD	•••• *	Password to use for generating the encryption key.
CONFIRM PASSWORD	•••• *	Must match password field.
AUTHENTICATION SERVICE	https://localhost:9031/demo-app-sp/go?acti *	The URL to which the user is redirected for an SSO event. This URL is part of an external application, which performs user authentication.



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

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.

Field	Description
Transport Mode	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.
Token Name	The name of the cookie or query parameter that contains the token. This name must be unique for each adapter instance.
Cipher Suite	The algorithm, cipher mode, and key size that should be used for encrypting the token.
Logout Service	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.
Cookie Domain	The server domain, preceded by a period (for example, .example.com). If no domain is specified, the value is obtained from the request.
Cookie Path	The path for the cookie that contains the token.
Token Lifetime	The duration (in seconds) for which the token is valid. Valid range is 1 to 28800.
Session Lifetime	The duration (in seconds) for which the token may be re-issued without authentication. Valid range is 1 to 259200.
Not Before Tolerance	The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.
Force SunJCE Provider	If selected, the SunJCE provider is forced for encryption/decryption.
Use Verbose Error Messages	If selected, use verbose TokenException messages.
Obfuscate Password	If selected, the password is obfuscated and password-strength validation is applied. Clearing the checkbox allows backward compatibility with previous OpenToken agents.
Session Cookie	If selected, OpenToken is set as a session cookie (rather than a persistent cookie). Applies only if Transport Mode is set as 'Cookie'.
Secure Cookie	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'.
Delete Cookie	If selected, the token cookie is deleted immediately after consumption. This option is valid only if Transport Mode is set to 'Cookie'.

Field	Description
Replay Prevention	Selecting this option is recommended only if Query Parameter is chosen as the token Transport Mode and POST is used by an associated connection to send the SAML assertion. If checked, PingFederate ensures that the token can be used only once. NOTE: Checking this option may affect resource utilization and performance.
Skip Malformed Attribute Detection	If not selected, it prevents insecure content from affecting the security of your application/agent. We recommend to update your applications with the latest version of the agent. We recommend not to change the value of this flag.

8. Click **Next**.

The screenshot shows the 'Create Adapter Instance' screen with the 'Actions' tab selected. A table lists the available actions:

ACTION NAME	ACTION DESCRIPTION	ACTION INVOCATION LINK
Download	Download the configuration file for the agent.	Invoke Download

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`, 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**.

The screenshot shows the 'Create Adapter Instance' screen with the 'Extended Contract' tab selected. The 'CORE CONTRACT' section is visible, showing the attribute 'subject'.

12. (Optional) On the Extended Contract screen, you can configure additional attributes for the adapter (see “[Extending an Adapter Contract](#)” on page 253).



Note: If this is a child instance, select the override checkbox to modify the configuration.

13. Click **Next**.

Main		Manage IdP Adapter Instances		Create Adapter Instance	
Type	IdP Adapter	Actions	Extended Contract	★ Adapter Attributes	Summary
As an IdP, some of your SP partners may choose to receive a pseudonym to uniquely identify a user. From the attributes in this authentication adapter, please select the values that you would like to use in constructing this unique identifier. Optionally, specify here any attributes that must be masked in log files.					
ATTRIBUTE	PSEUDONYM	MASK LOG VALUES			
subject	<input checked="" type="checkbox"/>	<input type="checkbox"/>			
<input type="checkbox"/> Mask all OGNL-expression generated log values.					

- 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 19).



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 25).

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 613).

- Click **Next**.
- On the Summary screen, review the configuration and click **Done**. You can also click any heading to go back and change information.
- On the Manage IdP Adapter Instances screen, click **Save**.



Important: You must click **Save** to retain the adapter configuration.

Configuring the SP OpenToken Adapter

- If you have not already done so, log on to the PingFederate administrative console and click **Adapters** under SP Configuration on the Main Menu.
- On the Manage Adapter Instances screen, click **Create New Adapter Instance**.

- 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.
- (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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.
- Click **Next**.

FIELD NAME	FIELD VALUE	DESCRIPTION
PASSWORD	<input type="password"/>	Password to use for generating the encryption key.
CONFIRM PASSWORD	<input type="password"/>	Must match password field.

Show Advanced Fields



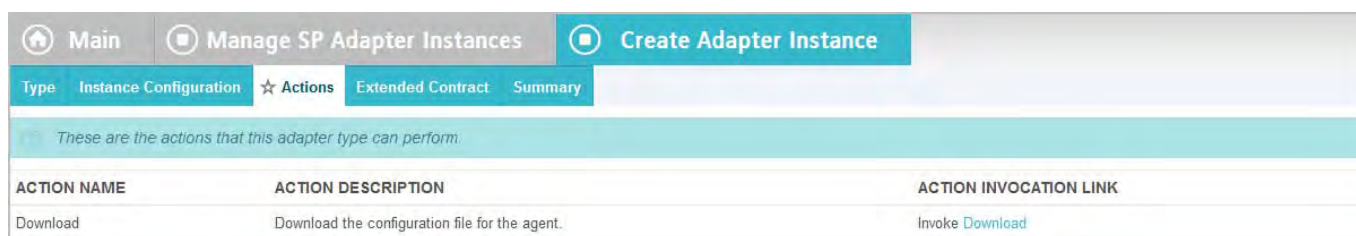
Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

- Enter values for the adapter configuration on the Instance Configuration screen.
These values are dependent on your developer’s implementation.
- (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.

Field	Description
Transport Mode	How the token is transported to/from the application, either via a query parameter, a cookie, or as a form POST (default). Note: Form POST is applicable only for an SP adapter instance.
Token Name	The name of the cookie or query parameter that contains the token. This name must be unique for each adapter instance.
Cipher Suite	The algorithm, cipher mode, and key size that should be used for encrypting the token.
Authentication Service	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.
Account Link Service	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.
Logout Service	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.
Cookie Domain	The server domain, preceded by a period (for example, <code>.example.com</code>). If no domain is specified, the value is obtained from the request.
Cookie Path	The path for the cookie that contains the token.
Token Lifetime	The duration (in seconds) for which the token is valid. Valid range is 1 to 28800.
Session Lifetime	The duration (in seconds) for which the token may be re-issued without authentication. Valid range is 1 to 259200.
Not Before Tolerance	The amount of time (in seconds) to allow for clock skew between servers. Valid range is 0 to 3600.
Force SunJCE Provider	If selected, the SunJCE provider is forced for encryption/decryption.
Obfuscate Password	If selected the password is obfuscated and password-strength validation is applied. Clearing the checkbox allows backward compatibility with previous OpenToken agents.
Session Cookie	If selected, OpenToken is set as a session cookie (rather than a persistent cookie). Applies only if Transport Mode is set as 'Cookie'.

Field	Description
Secure Cookie	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'.
Send Subject as Query Parameter	Selecting this checkbox sends the Subject ID as a clear-text query parameter, if Transport Mode is set to Query Parameter . If Transport Mode is set to Form POST , the Subject ID is sent as POST data.
Subject Query Parameter	The parameter name used for the Subject ID when the Send Subject ID as Query Parameter check box is selected.
Send Extended Attributes	Extended Attributes are typically sent only within the token, but this option overrides the normal behavior and allows the attributes to be included in browser cookies or query parameters.

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 365).



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 [“Configuring Target URL Mapping”](#) on page 366).

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 15).

PingFederate packages an HTTP Basic Adapter that delegates user authentication to a configured password credential validator (see [“Validating Password Credentials”](#) on page 235). 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 235).
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.

Adapter Instance Name and Id, select the Adapter Type, and a parent if applicable. The Adapter Type is limited to the ad installed on your server.

httpbasic *

httpbasic *

HTTP Basic IdP Adapter * Visit PingIdentity.com for additional types

None

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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

6. Click **Next**.



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 '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.

Property	Description
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.



Note: If this is a child instance, select the override checkbox to modify the configuration.

10. Click **Next**.

ATTRIBUTE	PSEUDONYM	MASK LOG VALUES
username	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Mask all OGNL-expression generated log values		



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 19).



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 25).

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 613).

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.

HTML Form 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 15).

PingFederate packages an HTML Form Adapter that delegates user authentication to a configured password credential validator (see [“Validating Password Credentials”](#) on page 235). 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.

PingFederate tracks login attempts per adapter instance. This capability adds a layer of protection against brute force and dictionary attacks. When the **Challenge Retries** threshold is reached, the user is locked out for one minute.

The default value for **Challenge Retries** is three. If you prefer a higher value, consider reviewing the account lockout policy of your directory servers first. For example, if the directory server's account lockout threshold is set to five and the **Challenge Retries** in the HTML Form Adapter is also set to five (or higher), the fifth single sign-on attempt could potentially lock the user accounts on the directory server.

The lockout period is customizable by modifying the **LockoutPeriod** value in `<pf_install>/pingfederate/server/default/data/config-store/`

`com.pingidentity.common.security.AccountLockingService.xml`
 1. This lockout is per adapter and is unlocked after the lockout period has lapsed.



Note: For a PingFederate cluster environment, modify the `com.pingidentity.common.security.AccountLockingService.xml` file on the console node and use the PingFederate administrative console to replicate the configuration to all engine nodes (see [Console Configuration Push](#) in the *Server Clustering Guide*).

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 235).
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.

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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

6. Click **Next**.

Complete the configuration necessary to hook up user security contexts in your environment. This configuration was designed into the adapter for use at your site.

CREDENTIAL VALIDATORS (A list of Password Credential Validators to be used for authentication.)

PASSWORD CREDENTIAL VALIDATOR INSTANCE Action

[Add a new row to 'Credential Validators'](#)

FIELD NAME	FIELD VALUE	DESCRIPTION
CHALLENGE RETRIES	3	Max value of User Challenge Retries.
SESSION STATE	<input checked="" type="radio"/> Globally <input type="radio"/> Per Adapter <input type="radio"/> None	Determines how state is maintained within one adapter or between different adapter instances.
SESSION TIMEOUT	60	Session Idle Timeout (in minutes). If left blank the timeout will be the Session Max Timeout. Ignored if 'None' is selected for Session State.
SESSION MAX TIMEOUT	480	Session Max Timeout (in minutes). Leave blank for indefinite sessions. Ignored if 'None' is selected for Session State.
LOGIN TEMPLATE	html form login template.html	HTML template (in <st_home>/server/default/conf/template) to render for login. The default value is html form login template.html.
LOGOUT PATH		Path on the PingFederate server to end a user's IDP session. Must include the initial slash (example: /mylogoutpath). (Resulting URL will be <url>/<st_host>/<port>/<st-Logout Path>). If specified, the path should be unique across all HTML Form IdP Adapter instances, including child instances.
LOGOUT REDIRECT		A fully qualified URL, usually at the SP, to which a user will be redirected after logout (applicable only when Logout Path is set above). When provided, this URL takes precedence over any Logout Template specified below.
LOGOUT TEMPLATE	idp logout success page template	HTML template (in <st_home>/server/default/conf/template) to render after logout (applicable only when Logout Path is set above and if Logout Redirect is not provided). The default value is idp logout success page template.html.
ALLOW PASSWORD CHANGES	<input type="checkbox"/>	Allows users to change their password using this adapter.
CHANGE PASSWORD TEMPLATE	html form change password template.html	HTML template (in <st_home>/server/default/conf/template) to render for a user to change their password. The default value is html form change password template.html.
CHANGE PASSWORD MESSAGE TEMPLATE	html form message template.html	HTML template (in <st_home>/server/default/conf/template) to render when a user is being redirected after successfully changing their password. If left blank, users are redirected without explanation. The default value is html form message template.html.
PASSWORD MANAGEMENT SYSTEM		A fully-qualified URL to your password management system where users can change their password. If left blank, password changes are handled by this adapter.
PASSWORD MANAGEMENT SYSTEM MESSAGE TEMPLATE	html form message template.html	HTML template (in <st_home>/server/default/conf/template) to render when a user is being redirected to the password management system to change their password. If left blank, users are redirected without explanation. The default value is html form message template.html.
LOGIN CHALLENGE TEMPLATE	html form login challenge template.html	HTML template (in <st_home>/server/default/conf/template) to render during a strong authentication as a second step. It is used to prompt the user to answer a challenge question after they login and the RADIUS Username Password Credential Validator is an example of where it could be used. The default value is html form login challenge template.html.
ENABLE 'REMEMBER MY USERNAME'	<input type="checkbox"/>	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.
'REMEMBER MY USERNAME' LIFETIME	30	Number of days that the username is stored. The cookie lifetime is reset upon each successful login in which the 'Remember My Username' checkbox is selected. The default is 30.

[Manage Password Credential Validators](#)



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 '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.

Property	Description
Challenge Retries	(Required) The account lockout threshold for this adapter instance. When the number of login failures reaches this threshold, the user is locked out for one minute. For more information, see the previous section, " HTML Form Adapter Configuration ".
Session State	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.
Session Timeout	The number of idle minutes before a session times out based on inactivity. The default value is 60 minutes (1 hour). If left blank, the lifetime falls back on the Session Max Timeout value. Ignored if None is selected for Session State. Note: This setting times out sessions based on inactivity.

Property	Description
Session Max Timeout	<p>The maximum lifetime (in minutes) before a session expires regardless if Session Timeout has been reached, or not. The default value is 480 minutes (8 hours). Ignored if None is selected for Session State.</p> <p>Note: This setting sets a maximum lifetime, subject to inactivity timeout (based on Session Timeout). Consider the following examples:</p> <p>A user initiated an SSO request at 9am and has not made another SSO request since then. At 10am, the session times out based on inactivity (based on Session Timeout's default value of 60 minutes).</p> <p>Another user initiated an SSO request at 9am and has been making SSO requests every hour at least once. This session does not time out because the user has been actively making SSO requests; however, the session does expire at 5pm (based on Session Max Timeout's default value of 8 hours).</p> <p>If you leave both Session Max Timeout and Session Timeout blank, sessions do not expire (until PingFederate restarts or the sessions are cleaned up by another means).</p> <p>If you leave Session Max Timeout blank but set a value for Session Timeout, sessions do not expire until they time out based on inactivity.</p> <p>Tip: Session information is stored in the PF cookie. 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 100.</p>
Login Template	<p>(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.</p>
Logout Path	<p>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.</p> <p>Note: If specified, the path should be unique across HTML Form IdP Adapter instances, including child instances.</p>
Logout Redirect	<p>The landing page at the SP after successful IdP logout (applicable only when Logout Path is set above).</p>
Logout Template	<p>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).</p>

HTML Form Adapter Configuration

Property	Description
Allow Password Changes	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. Note: 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).
Change Password Template	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.
Change Password Message Template	Template on the IdP server that notifies users their password was successfully changed.
Password Management System	URL for redirecting users to a company-specific password management system to change their password.
Password Management System Message Template	Template on the IdP server that notifies users they are being redirected to a password management system to change their password.
Login Challenge Template	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.
Enable 'Remember My Username'	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.
'Remember My Username' Lifetime	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.

10. Click **Next**.

- (Optional) On the Extended Contract screen, you can configure additional attributes for the adapter (see [“Extending an Adapter Contract”](#) on page 253).



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.

- Click **Next**.

ATTRIBUTE	PSEUDONYM	MASK LOG VALUES
username	<input type="checkbox"/>	<input type="checkbox"/>

Mask all OGNL-expression generated log values

- 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 19).



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 25).

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 613).

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 250).
2. Click **Adapters** under IdP Configuration on the Main Menu screen.
3. On the Manage Adapter Instances screen, click **Create New Adapter Instance**.

☆ Type IdP Adapter Extended Contract Adapter Attributes Summary

Enter an Adapter Instance Name and Id, select the Adapter Type, and a parent if applicable. The Adapter Type is limited to the adapters currently installed on your server.

Instance Name *

Instance Id *

Type * Visit Pingidentity.com for additional types

Parent Instance

- 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.

- (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 18). A child instance inherits the configuration of its parent unless overridden. You can specify overrides during the rest of the setup.

- Click **Next**.

Main
Manage IdP Adapter Instances
Create Adapter Instance

Type
★ IdP Adapter
Extended Contract
Adapter Attributes
Summary

Complete the configuration necessary to look up user security contexts in your environment. This configuration was designed into the adapter for use at your site.

A Composite Adapter allows existing adapter instances to be chained together to execute in sequence. Each configured instance of a Composite Adapter is treated as a single logical adapter instance.

ADAPTERS (Chained adapters)

ADAPTER INSTANCE	POLICY	AUTHN CONTEXT WEIGHT	AUTHN CONTEXT OVERRIDE	Action
HTML Form	<input checked="" type="radio"/> Required <input type="radio"/> Sufficient	3		Move down Edit Delete
Referenceld IdP Adapter	<input checked="" type="radio"/> Required <input type="radio"/> Sufficient	3		Move up Edit Delete
Add a new row to 'Adapters'				

ATTRIBUTE NAME SYNONYMS (Create synonyms between adapter attributes)

NAME	SYNONYM	Action
Add a new row to 'Attribute Name Synonyms'		

FIELD NAME	FIELD VALUE	DESCRIPTION
ATTRIBUTE INSERTION	<input checked="" type="radio"/> Add To Back <input type="radio"/> Add To Front *	Defines the order in which different values are returned for the same attribute name.



Note: If this is a child instance, select the override checkbox related to the settings you want to modify.

- 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](#)).

- 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 250).

- (Optional) Change or enter different defaults under any or all of the adjacent column headings under Adapters, as described in the following table.

Column	Description
Policy	<p>Required (the default) indicates authentication via this adapter instance is needed to continue SSO processing and invoke any remaining instances in the chain. If you are integrating multi-factor authentication, use this policy for each instance.</p> <p>Sufficient indicates that authentication via this adapter instance is enough to satisfy requirements (along with any required instances above). Any subsequent configured instances in the chain below are <i>not</i> invoked.</p> <p>Important: For the sufficient policy to work correctly, the adapter must return control to PingFederate after any kind of a failure. Currently, not all types of adapters have this capability. For an up-to-date list of adapters tested successfully under the Sufficient policy in failure mode, see Known Issues in the PingFederate Release Notes.</p>
AuthN Context Weight	<p>If more than one adapter instance in the chain is configured to return an authentication context, this relative weight is used to determine which value is included in the assertion—unless the value is overridden in the next column (see AuthN Context Override, below).</p> <p>If weights are the same for two or more contexts, the first one processed is included in the assertion.</p>
AuthN Context Override	<p>If provided, this value overrides any that may be returned from the adapter instance. The value in this field may be sent in the assertion if the associated adapter instance is invoked and its AuthN Context Weight is higher than other executed instances where authentication context is available.</p>

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

returned values are different, both values are sent for the synonym (see the next step).



Note: If this table is not used to identify synonymous attribute names, both names and their values are sent in the SAML assertion.

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**.

Main Manage IdP Adapter Instances Create Adapter Instance		
Type	IdP Adapter	Extended Contract
★ Adapter Attributes Summary		
As an IdP, some of your SP partners may choose to receive a pseudonym to uniquely identify a user. From the attributes in this authentication adapter, please select the values that you would like to use in constructing this unique identifier. Optionally, specify here any attributes that must be masked in log files.		
ATTRIBUTE	PSEUDONYM	MASK LOG VALUES
email	<input type="checkbox"/>	<input type="checkbox"/>
first_name	<input type="checkbox"/>	<input type="checkbox"/>
last_name	<input type="checkbox"/>	<input type="checkbox"/>
subject	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> Mask all OGNL-expression generated log values		



Note: If this is a child instance, select the override checkbox to modify the configuration.

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 19).



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 25).

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 613).

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 93).

SP endpoints also include those available for SCIM Inbound Provisioning (see [“Provisioning for SPs”](#) on page 36).

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 [“Adapter-to-Adapter Mapping”](#) on page 147) and retrieval of WS-Trust metadata (see [“WSC and WSP Support”](#) on page 8).

PingFederate provides a favorite icon for all Application Endpoints. For more information, see [“Customizing the Favicon for Application and Protocol Endpoints”](#) on page 101.

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:
`https://pingidentity.com:9031/idp/startSSO.ping.`



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](http://www.w3schools.com/tags/ref_urlencode.asp)” (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 591.

The following table shows the HTTP parameters for this endpoint.

Parameter	Description
<code>PartnerSpId</code> or <code>PARTNER</code>	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 <code>TargetResource</code> or <code>TARGET</code> (see below).
<code>TargetResource</code> or <code>TARGET</code> (optional)	For SAML 2.0, the value of either parameter is passed to the SP as the <code>RelayState</code> 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 <code>TARGET</code> . 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 264).
<code>InErrorResource</code> (optional)	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 93).

Parameter	Description
Binding (optional)	<p>Indicates the binding to be used; allowed values are URIs defined in the SAML specifications. For example, the SAML 2.0 applicable URIs are:</p> <pre>urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Artifact urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST</pre> <p>When the parameter is not used, the default ACS URL configured for the SP-partner connection is used, unless an ACS index is specified (see the next parameter, ACSIdx).</p>
ACSIdx (optional - SAML 2.0)	<p>Specifies the index number of partner's ACS (see "Setting Assertion Consumer Service URLs (SAML)" on page 314). Takes precedence over the Binding parameter if both are specified. If neither the binding nor index is specified in the call, the default ACS is used.</p>
IdpAdapterId (optional)	<p>Allows an application to call out what IdP adapter to use for authentication (in a configuration with multiple IdP adapters).</p> <p>Note: This parameter may be overridden by policy based on adapter selector configuration. For example, the CIDR Adapter Selector could enforce the use of a given adapter based on whether a user is on or off the network (see "Configuring Authentication Selectors" on page 255).</p>
RequestedFormat (optional - SAML 2.0)	<p>Allows control over the NameId format.</p>
vsid (optional)	<p>Specify the virtual server ID.</p> <p>When absent, PingFederate uses the default virtual server ID (if specified) for the connection (see "General Information" on page 276) or the SAML federation ID defined in Server Settings (see "Specifying Federation Information" on page 114).</p>

/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 167). 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.

Parameter	Description
TargetResource (optional)	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.
InErrorResource (optional)	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 93).
Binding (optional - SAML 2.0)	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 317).

/idp/writcdc.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.

Parameter	Description
TargetResource (optional)	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.

SP Endpoints

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)” (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 591.

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.

Parameter	Description
<code>PartnerIdpId</code> (required if more than one IdP connection is configured and <code>Domain</code> is not used)	The federation ID of the IdP that authenticates the user and issues an assertion. Note: This ID is case-sensitive.
<code>Domain</code> (required to invoke Auto-Connect)	The domain name associated with the requesting user’s IdP (see “ Using Auto-Connect ” on page 32). In this case, <code>PartnerIdpId</code> cannot be used.

Parameter	Description
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 372) or for individual connections (see “Configuring Default Target URLs (Optional)” on page 423), or both.
Binding (optional)	Indicates the binding to be used; allowed values are URLs defined in the SAML specifications. For example, the SAML 2.0 applicable URLs 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 416).
InErrorResource (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 93).
SpSessionAuthn AdapterId (optional)	The explicit SP adapter instance ID indicating the adapter to use to create an authenticated session or security context.
ForceAuthn (optional - SAML 2.0)	This parameter controls the attribute of the same name in the AuthnRequest. (The default is false.)
IsPassive (optional - SAML 2.0)	This parameter controls the attribute of the same name in the AuthnRequest. (The default is false.)
AllowCreate (optional - SAML 2.0)	Controls the value of the AllowCreate attribute of the NameIDPolicy element in the AuthnRequest. (The default is true.)
RequestedFormat (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.

Parameter	Description
AuthenticatingIdpId (optional - SAML 2.0)	This parameter indicates the preferred IdP for authenticating the user through an IdP proxy. The parameter specifies the value of the <code>IDPEntry ProviderID</code> attribute in the <code>Scoping</code> element of the <code>AuthnRequest</code> . Multiple values are permitted in order to build a preferred list.
RequestedACSIIdx (optional - SAML 2.0)	The index number of your site's Assertion Consumer Service, where you want the assertion to be sent.
RequestedAcsUrl (optional - SAML 2.0)	The URL of your site's Assertion Consumer Service, where you want the assertion to be sent.
RequestedBinding (optional - SAML 2.0)	Indicates the binding requested for the response containing the assertion; allowed values are URIs defined in the SAML specifications.
RequestedAuthnCtx (optional - SAML 2.0)	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.
RequestedAuthnDeclRef (optional - SAML 2.0)	An alternative to <code>RequestedAuthnCtx</code> , above, indicating the requested authentication context of the assertion by declaring any URI reference (see section 2.72.2 of the OASIS SAML document saml-core-2.0-os.pdf). Multiple values are permitted in order to build a preferred list.
RequestedSPNameQualifier (optional - SAML 2.0)	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 354).
vsid (optional)	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 383) or the SAML federation ID defined in Server Settings (see "Specifying Federation Information" on page 114).

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 [“Configuring Target URL Mapping”](#) on page 366).

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.

Parameter	Description
TargetResource (optional)	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.
Binding (optional - SAML 2.0)	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 IdP-partner connection is used (see “Specifying SLO Service URLs” on page 419).
InErrorResource (optional)	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 93).
SpSessionAuthn AdapterId (optional)	The SP adapter instance ID indicating which session to terminate and which IdP will receive the logout request.
SourceResource (optional)	A URL indicating the origin of the logout request. It is mapped to an adapter ID in order to designate which session to terminate.

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 it 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 567). 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.



Note: Some parameters described below can have multiple values. Specify these values by using multiple independent query string parameters of the same name.

Parameter	Description
Subject	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.

Parameter	Description
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 URIs (see section 8.3 of the SAML specifications (http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf). Note: For XASP, this parameter must be set to: <code>urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</code> If not specified, the parameter defaults to: <code>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</code> . 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 276) or the SAML federation ID defined in Server Settings (see "Specifying Federation Information" on page 114).

SCIM Inbound Provisioning Endpoints

The following sections describe the PingFederate endpoints for SCIM Inbound Provisioning (see ["Inbound Provisioning"](#) on page 36).

These endpoints are defined in the following SCIM 1.1 specifications:

- [SCIM Core Schema](http://www.simplecloud.info/specs/draft-scim-core-schema-01.html) (www.simplecloud.info/specs/draft-scim-core-schema-01.html)
- [SCIM Specification](http://www.simplecloud.info/specs/draft-scim-api-01.html) (www.simplecloud.info/specs/draft-scim-api-01.html)



Note: The PingFederate implementation is described in the next [couple](#) sections, [“/pf-scim/v1/Users”](#) and [“/pf-scim/v1/Groups”](#). Developers can find additional information about the implementation via the **Service Provider Configuration Endpoint** (see [“/pf-scim/v1/ServiceProviderConfigs”](#) on page 575).

/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 438). JSON is currently the only supported format for the HTTP message body.

HTTP Method	Description
POST	<p>Sends user attributes in JSON format—defined in the SCIM Core Schema—to create a new user.</p> <p>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 <code>id</code> attribute in the JSON response, and the full URL to reference the user is in the HTTP response Location header.</p> <p>For an existing user, you can also use POST either to update or delete/disable the user record by appending the user ID to the path (see the Note below) and setting the request header <code>X-HTTP-Method-Override</code> value to <code>PUT</code> or <code>DELETE</code>, respectively. (For more information, see the PUT and DELETE method descriptions below.)</p>
<p>Note: The following methods require a user ID parameter appended to the endpoint path: /pf-scim/v1/Users/<i>user_id</i></p>	
GET	<p>Retrieves all attributes for the specified user.</p> <p>A successful response is indicated by an HTTP 200 status code as well as the attributes.</p>

HTTP Method	Description
PUT	<p>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.</p> <p>A successful PUT operation returns an HTTP 200 status code and the entire updated user record within the response body.</p>
DELETE	<p>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 448).</p> <p>A successful response is indicated by an HTTP 200 status code.</p>
<p>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).</p>	

[/pf-scim/v1/Groups](#)

[The Groups endpoint is where client applications make HTTP requests to create, retrieve, update, and delete groups.](#)



Note: [Inbound provisioning for groups is optional. For more information, see “Choosing an IdP Connection Type” on page 381.](#)

[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 \(see “Configuring Inbound Provisioning” on page 438\). JSON is currently the only supported format for the HTTP message body.](#)

HTTP Method	Description
POST	<p>Sends group attributes in JSON format—defined in the SCIM Core Schema—to create a new group.</p> <p>If the group is successfully provisioned, the HTTP response indicates a 201 status code and contains a JSON body indicating all of the group attributes added to the datastore. The group ID is set as the <code>id</code> attribute in the JSON response, and the full URL to reference the group is in the HTTP response Location header.</p> <p>For an existing group, you can also use POST either to update or delete the group by appending the group ID to the path (see the Note below) and setting the request header <code>X-HTTP-Method-Override</code> value to <code>PUT</code> or <code>DELETE</code>, respectively. (For more information, see the PUT and DELETE method descriptions below.)</p>
<p>Note: The following methods require a group ID parameter appended to the endpoint path: /pf-scim/v1/Groups/<i>group id</i></p>	
GET	<p>Retrieves all attributes for the specified group.</p> <p>A successful response is indicated by an HTTP 200 status code as well as the attributes.</p>
PUT	<p>Updates group attributes for the specified group, 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.</p> <p>A successful PUT operation returns an HTTP 200 status code and the entire updated group record within the response body.</p>
DELETE	<p>Deletes the group record for the specified group.</p> <p>A successful response is indicated by an HTTP 200 status code.</p>
<p>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).</p>	

/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 106).

/pf/adapter2adapter.ping

This endpoint initiates direct IdP-to-SP adapter mapping, when that feature is configured (see “[Adapter-to-Adapter Mapping](#)” on page 147).

The following table shows the HTTP parameters for this endpoint.

Parameter	Description
TargetResource (optional)	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 372).
SpSessionAuthnAdapterId (optional)	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 “ Configuring Target URL Mapping ” on page 366).
IdpAdapterId (optional)	Indicates the IdP adapter to use for authentication if more than one IdP adapter is configured in adapter-to-adapter mappings.
InErrorResource (optional)	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 93).

/pf/sts.wst

This endpoint 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 “[Token Exchange Mapping](#)” on page 155).

The following table shows the HTTP parameters for this endpoint.

Parameter	Description
TokenProcessorId	Indicates the IdP token processor to use in the mapping. Required when multiple IdP token processors are configured in token-to-token mappings.
TokenGeneratorId	Indicates the SP token generator to use in the mapping. Required when multiple SP token generators are configured in token-to-token mappings.

/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:

Parameter	Description
PartnerSpId	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.
PartnerIdpId	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.
vsid (optional)	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 114).

/pf/federation_metadata.ping

This endpoint returns WS-Federation metadata.

The following table shows the HTTP parameters for this endpoint:

Parameter	Description
PartnerSpId	The Connection ID of the SP to whom the SAML token is issued. This parameter determines the connection for which metadata is generated.
PartnerIdpId	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.

Parameter	Description
vsid (optional)	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 114).



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:

- Construct a JSON object containing a key-value pair of the virtual server ID as follows:

```
{ "vsid": "<VirtualServerIdValue>" }
```

 Example: { "vsid": "Engineering" }
- Base64url-encode the JSON object.
 Example: **eyJ2c2lkIjoiRW5naW5lZXJpbmciQ**
- Insert the base64url-encoded value between /pf/ and /federation_metadata.ping (or /sts_mex.ping).
 Example: /pf/**eyJ2c2lkIjoiRW5naW5lZXJpbmciQ**/federation_metadata.ping (or /pf/**eyJ2c2lkIjoiRW5naW5lZXJpbmciQ**/sts_mex.ping)

For more information about base64url, see [RFC4648](https://tools.ietf.org/html/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

scheme (where the client identifier is the username, and the client secret is the password) or with the following HTTP request parameters:

Parameter	Description
client_id (optional)	The client identifier (see “Configuring a Client” on page 191).
client_secret (optional)	The client secret (as defined on the client management UI screen).

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:

Parameter	Description
grant_type (required)	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: <ul style="list-style-type: none">• <code>authorization_code</code>• <code>password</code>• <code>client_credentials</code>• <code>refresh_token</code>• <code>urn:ietf:params:oauth:grant-type:saml2-bearer</code>• <code>urn:pingidentity.com:oauth2:grant_type:validate_bearer</code> 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`.

Parameter	Description
code (required)	The authorization code received from the authorization server during the redirect interaction at the authorization endpoint when the <code>response_type</code> parameter is <code>code</code> (see “Endpoint: /as/authorization.oauth2” on page 586).

Parameter	Description
redirect_uri	<p>This parameter is required if the <code>redirect_uri</code> parameter was included in the authorization request that resulted in the issuance of the code (see “Authorization Endpoint” on page 586). The value here must match the authorization-request value, if applicable.</p> <p>The parameter is also required for clients with multiple redirect URIs or with one URI that uses wildcards (see “Configuring a Client” on page 191).</p>

Refresh Token Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `refresh_token`.

Parameter	Description
refresh_token (required)	The refresh token issued to the client during a previous access-token request.
scope (optional)	<p>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.</p> <p>Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 169).</p> <p>Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 190).</p>

Resource Owner Credentials (Password) Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `password`.

Parameter	Description
username (required)	The username, encoded as UTF-8.
password (required)	The password, encoded as UTF-8.
scope (optional)	<p>The scope of the access request. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 169).</p> <p>Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 190).</p>

Parameter	Description
validator_id (optional)	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.

Client Credentials Grant Type

These parameters apply when the `grant_type` parameter for `/as/token.oauth2` is set to `client_credentials`.

Parameter	Description
scope (optional)	The scope of the access request. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 169). Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 190).

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 579).

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`.

Parameter	Description
assertion (required)	A single SAML 2.0 assertion, which must be encoded using <code>base64url</code> , as described in Section 5 of RFC4648 (http://tools.ietf.org/html/rfc4648#section-5).
scope (optional)	The scope of the access request. Valid scope values are defined on the OAuth AS settings screen (see “Authorization Server Settings” on page 169). Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 190).

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`.

Parameter	Description
token (required)	The bearer access token to be validated.

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 177).

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"
  }
}
```

Access Token Management Parameters

access_token_manager_id (optional)

`access_token_manager_id` is the Instance ID of the desired access token manager. When specified, the PingFederate AS uses the desired access token

management instance for the request if it is eligible (see “[Access Token Management](#)” on page 175); otherwise it aborts the request.



Note: When the `access_token_manager_id` parameter is specified, the PingFederate AS ignores the `aud` parameter.

aud (optional)

`aud` is the resource URI the client wants to access. The provided value is matched against resource URIs configured in access token management instances (see “[Configuring Resource URIs](#)” on page 182). When a match is found, the PingFederate AS uses the corresponding access token management instance for the request if it is eligible (see “[Access Token Management](#)” on page 175); otherwise it aborts the request.

A match can be an exact match or a partial match where the provided URI has the same scheme and authority parts and a more specific path which would be contained within the path of the preconfigured resource URI. The PingFederate AS takes an exact match over a partial match. If there are multiple partial matches, the PingFederate AS takes the partial match where the provided URI matches more specifically against the preconfigured resource URI.

Example 1: A partial match

A Resource URI of `https://app.example.local` is a partial match for the following provided URIs:

- `https://app.example.local/file1.ext`
- `https://app.example.local/path/file2.ext`
- `https://app.example.local/path/more`

Example 2: An exact match is a better match than a partial match

Access Token Management Instances	Resource URIs
ATM1	<code>https://localhost:9031/app1</code> <code>https://localhost:9031/app2/data</code> <code>https://app.example.local</code>
ATM2	<code>https://localhost:9031/app1/data</code> <code>https://localhost:9031/app2/data/get</code>

`https://localhost:9031/app1` (a Resource URI preconfigured for ATM1) is a partial match for `https://localhost:9031/app1/data` (the provided URI). However, since `https://localhost:9031/app1/data` (a Resource URI preconfigured for ATM2) is an exact match against the provided URI, ATM2 is chosen.

Example 3: A more specific partial match is a better match

Both `https://localhost:9031/app2/data` (a Resource URI for ATM1) and `https://localhost:9031/app2/data/get` (a Resource URI for ATM2) are partial matches for `https://localhost:9031/app2/data/get/sample` (the provided URI). However, since `https://localhost:9031/app2/data/get` matches more specifically against the provided URI, ATM2 is chosen.

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:

Parameter	Description
client_id (optional)	The client identifier (see "Configuring a Client" on page 191).
client_secret (optional)	The client secret (as defined on the client management UI screen).

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:

Parameter	Description
token (required)	The token that the client wants to revoke.

Parameter	Description
token_type_hint (optional)	A hint about the type of token submitted for revocation. PingFederate supports the following values <ul style="list-style-type: none"> access_token refresh_token

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 Authentication Selector”](#) on page 260), 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:

Parameter	Description
response_type (required)	A value of <code>code</code> results in the Authorization Code grant type while a value of <code>token</code> implies the Implicit grant type. Additionally, a value of <code>id_token</code> can be requested by implicit clients.
response_mode (optional)	When set to <code>form_post</code> , the authorization response is returned to the client in an auto-POST form in accordance with the OAuth 2.0 Form Post Response Mode specification (openid.net/specs/oauth-v2-form-post-response-mode-1_0.html). <p>Note: At the time of writing, the OAuth 2.0 Form Post Response Mode specification is in draft status (version 03).</p>
client_id (required)	The client identifier (see “Configuring a Client” on page 191).

Parameter	Description
redirect_uri	<p>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 191). Optional for clients with only one specific redirect URI configured.</p> <p>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 580).</p>
scope (optional)	<p>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 169).</p> <p>Scopes can be restricted per client using the Client Management screen (see “Client Management” on page 190).</p>
state (optional)	<p>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.</p>
idp (or PartnerIdpId) (optional)	<p>A PingFederate OAuth AS parameter indicating the Entity ID/Connection ID of the IdP with whom to initiate Browser SSO for user authentication.</p>
pfidpadapterid (optional)	<p>A PingFederate OAuth AS parameter indicating the IdP Adapter Instance ID of the adapter to use for user authentication.</p> <p>Note: 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 Authentication Selectors” on page 255).</p>

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.

OpenID Connect Parameters

The table below displays OpenID Connect parameters for this endpoint:

Parameter	Description
nonce (optional)	<p>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.</p>

Parameter	Description
prompt (optional)	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.
acr_values (optional)	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.

Access Token Management Parameters

access_token_manager_id (optional)

`access_token_manager_id` is the Instance ID of the desired access token manager. When specified, the PingFederate AS uses the desired access token management instance for the request if it is eligible (see [“Access Token Management”](#) on page 175); otherwise it aborts the request.



Note: When the `access_token_manager_id` parameter is specified, the PingFederate AS ignores the `aud` parameter.

aud (optional)

`aud` is the resource URI the client wants to access. The provided value is matched against resource URIs configured in access token management instances (see [“Configuring Resource URIs”](#) on page 182). When a match is found, the PingFederate AS uses the corresponding access token management instance for the request if it is eligible (see [“Access Token Management”](#) on page 175); otherwise it aborts the request.

A match can be an exact match or a partial match where the provided URI has the same scheme and authority parts and a more specific path which would be contained within the path of the preconfigured resource URI. The PingFederate AS takes an exact match over a partial match. If there are multiple partial matches, the PingFederate AS takes the partial match where the provided URI matches more specifically against the preconfigured resource URI.

Example 1: A partial match

A Resource URI of `https://app.example.local` is a partial match for the following provided URIs:

- `https://app.example.local/file1.ext`
- `https://app.example.local/path/file2.ext`
- `https://app.example.local/path/more`

Example 2: An exact match is a better match than a partial match

Access Token Management Instances	Resource URIs
ATM1	https://localhost:9031/app1 https://localhost:9031/app2/data https://app.example.local
ATM2	https://localhost:9031/app1/data https://localhost:9031/app2/data/get

https://localhost:9031/app1 (a Resource URI preconfigured for ATM1) is a partial match for https://localhost:9031/app1/data (the provided URI). However, since https://localhost:9031/app1/data (a Resource URI preconfigured for ATM2) is an exact match against the provided URI, ATM2 is chosen.

Example 3: A more specific partial match is a better match

Both https://localhost:9031/app2/data (a Resource URI for ATM1) and https://localhost:9031/app2/data/get (a Resource URI for ATM2) are partial matches for https://localhost:9031/app2/data/get/sample (the provided URI). However, since https://localhost:9031/app2/data/get matches more specifically against the provided URI, ATM2 is chosen.

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 functionality. The grants displayed are those associated with the `USER_KEY` of the authenticated user. The same attribute mapping(s) from the authentication source to `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.



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.

Parameter	Description
idp (or PartnerIdpId) (optional)	Indicates the Entity ID/Connection ID of the IdP with whom to initiate Browser SSO for user authentication.

Parameter	Description
pfidpadapterid (optional)	Indicates the IdP Adapter Instance ID of the adapter to use for user authentication. Note: 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 Authentication Selectors” on page 255).

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 579).

OpenID Connect Metadata Endpoint

This public endpoint provides metadata, such as the UserInfo endpoint, needed for an OAuth client to interface with PingFederate using the OpenID Connect protocol.



Tip: The UserInfo endpoint is where OAuth clients send access tokens to for the purpose of retrieving additional claims about the users. For more information, see [“OpenID Connect”](#) on page 15.

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.



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 84).

- [SSO Directory Service](#) — Provides Web application developers with information regarding partner connections and adapter instances.



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 233.

PingFederate also provides the following REST-based Web Services:

- [OAuth Client Management Service](#) — Useful for managing OAuth client applications, where needed (see [“About OAuth”](#) on page 10 and [“Client Management”](#) on page 190).
- [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.

- [PingFederate Administrative API](#) — Allows developers and non-developers to change PingFederate IdP Connections settings (see “[Service Provider SSO Configuration](#)” on page 361).

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.



Note: If you do not want to allow use of the service, it should not be deployed: remove the WAR file from the `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 269.

For information about exporting IdP connection at an SP site, see “[From the Manage Connections Screen](#)” on page 378.

Using the Connection Service

The Connection Web Service exposes the following method for exporting connections:

```
public string getConnection(
    String entityId,
    String role,) throws IOException
```

Code Sample

```
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 via the PingFederate administrative console (see [“Accessing Connections”](#) on page 268 for SP connections or [“Accessing IdP Connections”](#) on page 377 for IdP connections). You can also import the connections 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:

```
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:

```
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\"
...
</EntityDescriptor>";
boolean allowUpdate = false;
call.invoke(new Object[]{xml, allowUpdate});
```

Deleting Connections

The Web Service exposes the following method for connection deletion:

```
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:

```
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



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 597).

- `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 233). We recommend the use of HTTPS when including credentials.

```
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 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 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>
    <getIDPListResponse
      soapenv:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/">
      <getIDPListReturn
        soapenc:arrayType=
          "ns1:IDP[2]" xsi:type="soapenc:Array"
        xmlns:ns1="urn:BeanService"
        xmlns:soapenc=
          "http://schemas.xmlsoap.org/soap/encoding">
        <getIDPListReturn href="#id0" />
        <getIDPListReturn href="#id1" />
      </getIDPListReturn>
    </getIDPListResponse>
    <multiRef id="id0" soapenc:root="0"
      soapenv:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/"
      xsi:type="ns2:IDP"
      xmlns:soapenc=
        "http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns2="urn:BeanService">
      <company xsi:type="xsd:string">MegaMarket</company>
      <entityId xsi:type="xsd:string">www.megamarket.com
      </entityId>
    </multiRef>
    <multiRef id="id1" soapenc:root="0"
      soapenv:encodingStyle=
        "http://schemas.xmlsoap.org/soap/encoding/"
      xsi:type="ns3:IDP" xmlns:ns3="urn:BeanService"
      xmlns:soapenc=
        "http://schemas.xmlsoap.org/soap/encoding/">
      <company xsi:type="xsd:string">Ping</company>
      <entityId
        xsi:type="xsd:string">pingfederate3:default:entityId
      </entityId>
    </multiRef>
  </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 190).



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 137.



Tip: PingFederate administrative API can also be used to manage OAuth clients programmatically regardless if the client records are managed in XML files or in a database. For more information, see [“PingFederate Administrative API”](#) on page 610.

Endpoints

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 235). The configured Credential Validator, in turn, must be selected in the OAuth AS configuration (see [“Authorization Server Settings”](#) on page 169).

/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 190.

The required MIME type is `application/json`.

JSON Parameters

Table 23

Parameter	Description
clientId	(Required) A unique ID for the client.
name	(Required) A descriptive name for the client.
refreshRolling	Indicates whether a new refresh token is issued with each new access token (see “Field Descriptions” on page 193). Allowed values: <code>true</code> or <code>false</code> . When not provided, the global setting for the AS is used (see “Authorization Server Settings” on page 169).
redirectUri	The URI(s) to which the OAuth AS redirects the resource owner’s user agent after authorization is obtained. <i>Required</i> for Implicit and Authorization Code grant types (see <code>grantTypes</code> below).
logoUrl	A URL for the logo presented to the user on the grant revocation page.
secret	Client password or phrase, <i>required</i> for the Client Credentials grant type unless client-certificate authentication is needed (see next two parameters).
clientCertIssuerDn	Client TLS certificate issuer, <i>required</i> for the Client Credentials grant type unless a <code>secret</code> is provided.
clientCertSubjectDn	Client TLS certificate subject, <i>required</i> for the Client Credentials grant type unless a <code>secret</code> is provided.
description	A description of what the client application does, displayed in browser when the user is prompted for authorization.
persistentGrantExpirationType	Indicates whether to override the global setting for the AS (see “Authorization Server Settings” on page 169). Allowed values: <ul style="list-style-type: none"> • <code>SERVER_DEFAULT</code> (the default) – Use the global setting for the AS. • <code>NONE</code> – Grants do not expire, regardless of the global setting. • <code>OVERRIDE_SERVER_DEFAULT</code> – Use with both of the <code>persistentGrant*</code> parameters below to set the expiration time period.

Table 23

Parameter	Description
<code>persistentGrantExpirationTime</code>	An integer representing units of time for storage of persistent grants for this client—use with <code>persistentGrantExpirationTimeUnit</code> (see below) and only when <code>persistentGrantExpirationType</code> is set to <code>OVERRIDE_SERVER_DEFAULT</code> .
<code>persistentGrantExpirationTimeUnit</code>	Units for the expiration time set in the parameter above (if applicable). Allow values: h – hours d – days
<code>bypassApprovalPage</code>	If set to <code>true</code> , user consent to resource access is assumed and the approval page is not presented.
<code>restrictScopes</code>	If set to <code>true</code> , limits client access to a subset of the scopes defined for the AS (see “Authorization Server Settings” on page 169). Scopes are limited to the default scope and any listed for <code>restrictedScopes</code> (see below). If no scopes are listed and this parameter is <code>true</code> , only the default scope is available for the client.
<code>restrictedScopes</code>	When used with <code>restrictScopes</code> , limits access to the scope(s) provided in the JSON list, in addition to the default scope.
<code>grantTypes</code>	One or more grant types allowed for the client (see “Grant Types” on page 11). Allowed values: <ul style="list-style-type: none"> • <code>authorization_code</code> • <code>password</code> (Resource Owner Password Credentials) • <code>refresh_token</code> (Use with <code>authorization_code</code> and/or <code>password</code> grant types.) • <code>client_credentials</code> • <code>implicit</code> • <code>extension</code> (SAML 2.0 Bearer) • <code>urn:pingidentity.com:oauth2:grant_type:validate_bearer</code> (Access Token Validation) <p>Notes:</p> <ul style="list-style-type: none"> • At least one grant type is required. • Separate multiple values with commas.

Table 23

Parameter	Description
defaultAccessTokenManagerId	The default access token manager for this client (see “Access Token Management” on page 175).
idTokenSigningAlgorithm	The JSON Web Signature (JWS) algorithm required for the ID Token. Allowed values: <ul style="list-style-type: none"> • none - No signing algorithm • HS256 - HMAC using SHA-256 • HS384 - HMAC using SHA-384 • HS512 - HMAC using SHA-512 • RS256 - RSA using SHA-256 • RS384 - RSA using SHA-384 • RS512 - RSA using SHA-512 • ES256 - ECDSA using P256 Curve and SHA-256 • ES384 - ECDSA using P384 Curve and SHA-384 • ES512 - ECDSA using P521 Curve and SHA-512
policyGroupId	The desired Open ID Connect policy.
grantAccessSessionRevocationApi	If set to <code>true</code> , this client is allowed to access the Session Revocation API (see “Back-Channel Session Revocation” on page 168).
pingAccessLogoutCapable	If set to <code>true</code> , 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 167).
logoutUris	A list of client logout URI's which will be invoked when a user logs out through one of PingFederate's SLO endpoints (see “Asynchronous Front-Channel Logout” on page 167). Similar to <code>redirectedUris</code> , multiple entries are allowed.

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"
]
}
1}

```

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",
"redirectUri": [
  "http://www.url.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"
]
}
1}
```

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.
- 500 – Internal Server Error
An unknown error has occurred.

/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 235). The configured Credential Validator, in turn, must be selected in the OAuth AS configuration (see [“Authorization Server Settings”](#) on page 169).

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 137.

Parameters

clientId	The client identifier to retrieve or revoke grants for (see “Configuring a Client” on page 191).
userKey	The user key to retrieve or revoke grants for. The user key is defined under IdP Adapter Mapping and Resource Owner Credentials Mapping.
grantId (optional)	Access grant identifier used to retrieve or revoke a specific grant. The value corresponds to the <code>id</code> field of the JSON array of grants returned from a previous GET <code>/oauth/clients/{clientId}/grants</code> or GET <code>/oauth/users/{userKey}/grants</code> . If this parameter is not specified, the request applies to all grants for the client or user.

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:

```
{
  "items": [{
    "id": "gn3T3qGVjoHL9p8HKtCSZNriwg9H3DA8",
    "userKey": "joe",
    "grantType": "IMPLICIT",
    "scopes": [],
    "clientId": "im_client",
    "issued": "2014-03-26T21:15:38.551Z",
    "updated": "2014-03-26T21:15:38.551Z"
  }]
}
```

Returns

- 200 – Success
- 204 – Success with no content returned
Will be seen when revoking an access grant
- 401 – Invalid Credentials.
The user does not exist or the password is incorrect.
- 404 – Not found
Resource (user, client, or access grant) not found
- 500 – Internal Server Error
An unknown error has occurred.

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 168).



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 191).

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 590). 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 184). 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:

```
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 191).

Sample GET request and responses

A GET request to query the revocation status for a session with a session identifier of `abc123`:

```
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 610.

PingFederate records actions performed via the administrative API in the `admin-api.log` file (see “[Administrator API Audit Logging](#)” on page 53).

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!**

The Request URL, Response Body, Response Code, and Response Headers appear.



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:

```
<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 35)
- The Show Advanced Criteria section on the Issuance Criteria screen following each of the administrative-console Fulfillment screens

ATTRIBUTE CONTRACT	SOURCE	VALUE	ACTIONS
SAML_SUBJECT	Adapter	subject	None available
email	Adapter	email	None available
fname	- SELECT - Adapter Context	fname	None available
Iname	Expression	Iname	None available
	Text		

Figure 5: Attribute Contract Fulfillment (Example)

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 618).

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 618.

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("(?i)
.*CN=Contractor,cn=users,dc=company,dc=com.*|
.*CN=Employee,cn=users,dc=company,dc=com.*")
:@java.lang.Boolean@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.

The following expression extracts the domain information out of an email address (mail) and returns true if it matches a specific domain (company.com):

```
#this.get("mail")!=null?
(
  #email=#this.get("mail").toString(),
  #atSign="@",
  #at=#mail.indexOf(#atSign),
  #at > 0?
  (
    #domain=#mail.subject(#at+1),
    #domain.matches("(?i)company.com")
  ):false
):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 39), 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](https://commons.apache.org/ognl/index.html) (`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 25).



Note: The test function does not work for the `context.httpRequest` attribute, because its value is an object rather than text.

Here is an example of the edit screen, from the IdP configuration flow:

The screenshot shows the 'IdP Adapter Mapping' configuration screen. At the top, there are navigation tabs: Main, SP Connection, Browser SSO, Assertion Creation, and IdP Adapter Mapping. Below these are sub-tabs: Adapter Instance, Assertion Mapping, Attribute Sources & User Lookup, Failsafe Attribute Source, Attribute Contract Fulfillment (selected), and Summary. A message states: 'Fulfill your Attribute Contract with values from the authentication adapter or dynamic text values. These values will be used if the user cannot be located in data stores. Coordinate with your partner to determine what default values to send.'

ATTRIBUTE	VALUE	TEST EXPRESSION	ACTIONS
context.AuthenticationCtx	<input type="text"/>	<div style="border: 1px solid #ccc; height: 150px; width: 100%;"></div>	Update / Cancel
context.ClientIp	<input type="text"/>		
context.HttpRequest	<input type="text"/>		
context.TargetResource	<input type="text"/>		
email	<input type="text"/>		
fname	<input type="text"/>		
lname	<input type="text"/>		
sessionId	<input type="text"/>		
subject	<input type="text"/>		
username	<input type="text"/>		

Below the table, there is a section for 'TEST RESULTS' and an 'ACTIONS' button labeled 'Test'.

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.

Troubleshooting

Basic troubleshooting tips are provided here to help overcome common difficulties. Help is also available from the [Support Center](https://pingidentity.com/support-center) at pingidentity.com.

This appendix contains the following sections:

- “[Data Store Issues](#)” on page 621
- “[Installation Issues](#)” on page 622
- “[Runtime Issues](#)” on page 622
- “[Server Startup](#)” on page 622

Data Store Issues

Table 24: Troubleshooting Data Stores

Problem	Solution
When setting up the JDBC data store, a connection cannot be established.	Verify that the proper drivers and connectors have been installed. Also, verify the connection URL, username, and password. If unsuccessful, contact your database administrator.
Cannot connect to a Directory Service with the LDAP protocol.	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 222).

Installation Issues

Table 25: Troubleshooting Installation

Problem	Solution
Error message "Not enough memory on the server"	Verify that there is at least 1,024 MB of RAM installed on the server (see " System Requirements " in the "Installation" chapter of <i>Getting Started</i>).

Runtime Issues

Table 26: Troubleshooting Runtime Issues

Problem	Solution
Certificates unexpectedly expire.	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 105).
Receiving CrossModule/ Network Errors	Verify network connections to the Hardware Security Modules (HSMs) are active and running. Also ensure the HSMs have not been unintentionally shut down.

Server Startup

Table 27: Troubleshooting the PingFederate Server

Problem	Solution
PingFederate does not start.	Make sure that the Java SDK is installed (see " Installing the JDK " in the "Installation" chapter of <i>Getting Started</i>).
The server starts but indicates the license file is not found or invalid.	Ensure a current license is installed (see " Installing PingFederate " on page 17). If the server is part of a cluster, make sure the license is installed for only one server node, then restart all nodes.

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 of 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 15).

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 method of encryption.

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.

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 a 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.

<RequestSecurityToken>

(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.

<RequestSecurityTokenResponse>

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

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.

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

ACS	Assertion Consumer Service	LDAP	Lightweight Directory Access Protocol
API	Application Programmer Interface	NAS	Network Access Server
ARS	Artifact Resolution Service	O	Organization
CA	Certificate Authority	OASIS	Organization for the Advancement of Structured Information Standards
CRL	Certificate Revocation List	OCSP	Online Certificate Status Protocol
CSR	Certificate Signing Request	OU	Organizational Unit
DBMS	Database Management System	PKI	Public Key Infrastructure
DMZ	Demilitarized Zone	RADIUS	Remote Authentication Dial-in User Service
DN	Distinguished Name (certificate identifier)	RDBMS	Relational Database Management System
DNS	Domain Name System	RST	<RequestSecurityToken>
EIM	Enterprise Identity Management	RSTR	<RequestSecurityTokenResponse>
GUI	Graphical User Interface	SAML	Security Assertion Markup Language
HTTP	HyperText Transfer Protocol	SaaS	Software as a Service
HTTPS	Secure HyperText Transfer Protocol	SCIM	System for Cross-domain Identity Management
IdM	Identity Management	SDK	Software Development Kit
IdP	Identity Provider	SP	Service Provider
IP	Internet Protocol		
J2SDK	Java 2 Software Development Kit		
JDBC	Java Database Connectivity (JDBC)		

List of Acronyms

SLO	Single Logout
SOA	service-oriented architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
SSL/TLS	Secure Sockets Layer/Transport Level Security
SSO	Single Sign-On
SSTC	Security Services Technical Committee (of OASIS)
STS	Security Token Service
TCP	Transmission Control Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WCF	Windows Communication Foundation
WIF	Windows Identity Foundation
WSC	Web Service Client
WSP	Web Service Provider
WSS	Web Services Security
XASP	X.509 Attribute Sharing Profile
XML	Extensible Markup Language

PingFederate[®]

Server Clustering Guide



© 2006-2015 Ping Identity® Corporation. All rights reserved.

PingFederate *Server Clustering Guide*
Version 7.3
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909
Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: **January 28, 2015**.

Contents

Overview	4
General Architecture	4
Group RPC-Oriented Approach.....	4
Load Balancing.....	4
Server Modes	5
Deploying Clustered Servers	5
Configuring Startup Properties	6
Managing Deployment Architecture.....	8
Sharing All Nodes	10
Designating State Servers.....	11
Defining Subclusters	12
Runtime State-Management Services	13
Inter-Request State Management Service	13
SP Session Registry Service.....	14
IdP Session Registry Service	15
Assertion Replay Prevention Service.....	15
Artifact-Message Persistence and Retrieval Service.....	16
Back-Channel Session Revocation Service	18
Extensibility and Other Services	18
Deploying Provisioning Failover	19
Configuring Runtime Properties.....	20
Synchronizing Server Runtime Configuration	20
Console Configuration Push.....	21
Configuration-Archive Deployment	22

Overview

PingFederate provides clustering features that allow a group of PingFederate servers to appear to browsers and partner federation servers as a single system. In this configuration, all client traffic normally goes through a load balancer, which routes requests to the PingFederate servers in the cluster. User-session states and configuration data are shared among the servers, enabling them to process requests as a single entity.

Tip: PingFederate provides separate failover capabilities specifically for Outbound Provisioning, which by itself does not require either load balancing or state management (see [Deploying Provisioning Failover](#) on page 19).

When deployed appropriately, server clustering can facilitate high availability of critical services. Clustering can also increase performance and overall system throughput. It is important to understand, however, that availability and performance are often at opposite ends of the deployment spectrum. Thus, you may need to make some configuration tradeoffs that balance availability with performance to accommodate specific deployment goals. Some of these choices are identified throughout this *Guide*.

General Architecture

PingFederate abstracts its runtime session-state management behind Java service interfaces. This enables PingFederate to use interface implementations without regard to underlying storage and sharing mechanisms. The abstraction also provides a well-defined point of extensibility in PingFederate (see [Runtime State-Management Services](#) on page 13).

Group RPC-Oriented Approach

The prepackaged state-management implementations are designed to accommodate a variety of deployments. The implementations leverage a remote-procedure-call (RPC) framework for reliable group communication, allowing PingFederate servers within a cluster to share state information.

Load Balancing

Clustered deployments of PingFederate for single sign-on (SSO) and logout transactions require the use of at least one load balancer, fronting multiple PingFederate servers.

When a client accesses the load balancer's virtual IP, the balancer distributes the request to one of the PingFederate servers in the cluster. The method that the balancer uses to select the appropriate server can vary from simple to highly complex, depending on deployment requirements. Specific balancing strategies, their strengths and weaknesses, as well as the impacts on PingFederate are discussed later (see [Managing Deployment Architecture](#) on page 8).

The PingFederate software distribution does not contain a load balancer. Numerous hardware or software products are available commercially, including free downloads.

Load balancers may incorporate SSL/TLS accelerators or work closely with them. Due to the high computational overhead of the SSL handshake, Ping Identity recommends terminating SSL/TLS on a

dedicated server external to PingFederate for deployments in which performance is a concern. You can still use SSL between the proxy or balancer and PingFederate, but as a separate connection.

Server Modes

In a cluster, you can configure each PingFederate instance, or *node*, as either an administrative console or a runtime engine. Runtime engines service federated-identity protocol requests, while the console server administers policy and configuration for the entire cluster (via the administrative console). A cluster may contain one or more runtime nodes but only one console node (see the next section).

Note: The PingFederate administrative console node must be configured to run outside of the load-balanced group to successfully process SSO requests.

Deploying Clustered Servers

Follow the steps below to configure and deploy clustered PingFederate servers.

Note: Additional steps are required to set up failover for provisioning. Alternatively, if you are grouping servers *exclusively* to provide for provisioning failover, skip this section and refer to information under [Deploying Provisioning Failover](#) on page 19.

To configure PingFederate servers in a cluster:

1. For each node in a cluster, follow the PingFederate installation procedures (see Installation in *Getting Started*).

Tip: You need only install one license file on any one machine in the cluster; the key will be pushed out to the other servers.

2. For each server instance, edit clustering properties in the `run.properties` file located in the `<pf_install>/pingfederate/bin` directory (see the next section, [Configuring Startup Properties](#)).
3. (Optional) Adjust configuration files that control cluster architecture and runtime session-state management (see [Managing Deployment Architecture](#) on page 8 and [Runtime State-Management Services](#) on page 13).
4. (Optional) If Outbound Provisioning is configured at your site and you want to provide failover capabilities, identify provisioning failover nodes (see [Deploying Provisioning Failover](#) on page 19).
5. Start or restart all of the PingFederate servers.

Refer to Starting and Stopping PingFederate, in the PingFederate *Administrator's Manual*, if needed.

Important: Start the server containing the license file before starting the other servers.

6. After you configure (or reconfigure) PingFederate through the administrative console, replicate the configuration on all nodes in the cluster (see [Synchronizing Server Runtime Configuration](#) on page 20).

The *Administrator's Manual* and installed **Help** provide detailed information on using the administrative console.

Configuring Startup Properties

The `run.properties` in the `<pf_install>/pingfederate/bin` directory contains startup clustering configuration options. Configure the properties described in the following table according to your needs for each node in the cluster.

Note: This table does not include information about properties used for provisioning failover (see [Deploying Provisioning Failover](#) on page 19).

Property	Description
<code>pf.operational.mode</code>	Controls the operational mode of the PingFederate server. Refer to the properties file for a list of valid values and descriptions. Note that the value <code>STANDALONE</code> should not be used in a cluster unless user-state management is not needed for any reason and configuration-archive deployment is used as the configuration synchronization method (see Synchronizing Server Runtime Configuration on page 20).
<code>pf.cluster.node.index</code>	Each server in a cluster must have a unique index number, which is used to identify peers and optimize inter-node communication. (Range: 0-65535) If no value is set for the node index, the system assigns a default index derived from the last two octets of the IP address. We recommend, however, that you assign static indexes. Tip: The PingFederate Cluster Node Selector, available in version 7.0 and higher, provides the capability of overriding adapter processing based on the runtime node servicing a request.
<code>pf.cluster.auth.pwd</code>	Sets the password that each node in the cluster must use to authenticate when joining the group. This prevents unauthorized nodes from joining a cluster. (Value: any string, or blank) All nodes in a cluster must have the same value set for this property. The first node to start up will set the password for the cluster.
<code>pf.cluster.encrypt</code>	Indicates whether or not to encrypt network traffic sent between nodes in a cluster. (Values: <code>true</code> <code>false</code> [default]) When set to <code>true</code> , communication within the cluster is encrypted with a symmetric key derived from the value of the <code>pf.cluster.auth.pwd</code> property. All nodes in a cluster must have the same value set for this property.

Property	Description
pf.cluster.bind.address	(Required) Defines the IP address of the network interface to which the group communication should bind. For machines with more than one network interface, you can use this property to increase performance (particularly with UDP) as well as improve security by segmenting group-communication traffic onto a private network or VLAN.
pf.cluster.bind.port	(Required) The port associated with the bind-address property above, or with the default network interface used.
pf.cluster.failure.detection.bind.port	Indicates the bind port of a server socket that is opened on the given node and used by other nodes as part of one of the cluster's failure-detection mechanisms. If zero or unspecified, a random available port is used.
pf.cluster.transport.protocol	<p>Indicates the transport protocol used for group communication (values: <code>udp</code> <code>tcp</code>).</p> <p>Use UDP multicast when IP multicasting is enabled in the network environment and the majority of group traffic is point-to-full-group. You must also provide values for the <code>pf.cluster.mcast*</code> properties described below.</p> <p>Use TCP for geographically dispersed servers or when multicast is not available or disabled for some other reason (routers discard multicast messaging). TCP may also be appropriate if your cluster configuration employs more point-to-point or point-to-few messaging than point-to-group. You must also provide a value for the <code>pf.cluster.tcp.discovery.inital.hosts</code> property described below.</p> <p>Note: This property is a reference to a protocol-stack XML configuration file located in the <code><pf_install>/pingfederate/server/default/conf/</code> directory. Two stacks are provided: one for UDP multicast and one for TCP. Administrators can customize either stack or add to it as needed by modifying the associated configuration file.</p>
pf.cluster.mcast.group.address	<p>Defines the IP address shared among nodes in the same cluster for UDP multicast communication; required when UDP is set as the transport protocol. (Range: 224.0.0.0 to 239.255.255.255; note that some addresses in this range are reserved for other purposes.) This property is not used for TCP.</p> <p>All nodes in a cluster must use the same address for this property and the port property below.</p>
pf.cluster.mcast.group.port	Defines the port associated with the property above, <code>pf.cluster.mcast.group.address</code> .

Property	Description
pf.cluster.tcp.discovery.initial.hosts	<p>Designates the initial hosts to be contacted for group membership information when discovering and joining the group; required when TCP is set as the transport protocol. The value is a comma-separated list of host names (or IPs) and ports.</p> <p>Example: <code>host1[7600],10.0.1.4[7600],host7[1033],10.0.9.45[2231]</code></p> <p>Discovering and managing group membership is more difficult using TCP, which does not provide the built-in group semantics of IP multicast. Therefore, at least one of the members of the group must be known in advance and statically configured on each node. It is recommended that as many hosts as possible be included for this property on each cluster node, to increase the likelihood of new members finding and joining the group.</p>

Managing Deployment Architecture

In a cluster consisting of a large number of nodes, it may be desirable to share data with only a subset of nodes (for performance reasons). To facilitate this, most of the group RPC-based service implementations (see [Runtime State-Management Services](#) on page 13) make use of a *preferred-nodes* concept, which allows each node to have a list of other nodes (identified by index) with which it shares data.

Each service implementation is controlled separately by a configuration file located in the `<pf_install>/pingfederate/server/default/conf` directory. Any changes must be replicated manually for each cluster node. The table below indicates the configuration file that applies to each implementation. Refer to the indicated sections in this *Guide* for detailed information about each implementation.

Configuration File	RPC-Based Service Implementation
cluster-artifact.conf	Artifact-Message Persistence and Retrieval Service
cluster-assertion-replay-prevention.conf	Assertion Replay Prevention Service
cluster-idp-session-registry.conf	IdP Session Registry Service
cluster-inter-request-state.conf	Inter-Request State Management Service
cluster-session-revocation.conf	Back-Channel Session Revocation Service
cluster-sp-session-registry.conf	SP Session Registry Service

The following table describes properties contained in the configuration files (the Artifact-Message Persistence and Retrieval Service uses only `rpc.timeout`):

Property	Description
preferred.node.indices	A comma-separated list of indices identifying the nodes with which this server shares session-state data for the associated service. If blank, all nodes in the cluster are used.
rpc.timeout	How long, in milliseconds, the server waits before timing out unresponsive RPC invocations.
synchronous.retrieve.majority.only	Indicates how many responses to wait for when making synchronous RPC calls (values: <code>true</code> <code>false</code>). When <code>true</code> (the default), the server waits for the majority of recipients to respond. When <code>false</code> , it waits for all recipients to respond.
bulk.revoked.sris.timeout (found only in cluster-session-revocation.conf)	A node downloads a full revocation list from another node during startup or when it rejoins a cluster after disconnected from it (possibly due to a temporary network issue). This setting determines the amount of time (in milliseconds) PingFederate waits before aborting the download and reporting a timeout error. The default value is 10000 milliseconds (10 seconds).
read.local.only (found only in cluster-session-revocation.conf)	<p>Determines whether PingFederate should process queries for revocation status by searching the local revocation list or collecting the information from other engine nodes in the cluster.</p> <p>When <code>true</code> (the default), queries for revocation status are processed locally. When <code>false</code>, the processing node pulls revocation status from other engine nodes in the cluster (subject to the <code>rpc.timeout</code> value).</p> <p>Note: When adding a session to the revocation list, the processing node always propagates the information to all engine nodes in the cluster (see Back-Channel Session Revocation Service on page 18).</p>

Note that within a single cluster deployment, individual services can use different preferred nodes. In other words, you can set different values for the `preferred.node.indices` property for each service.

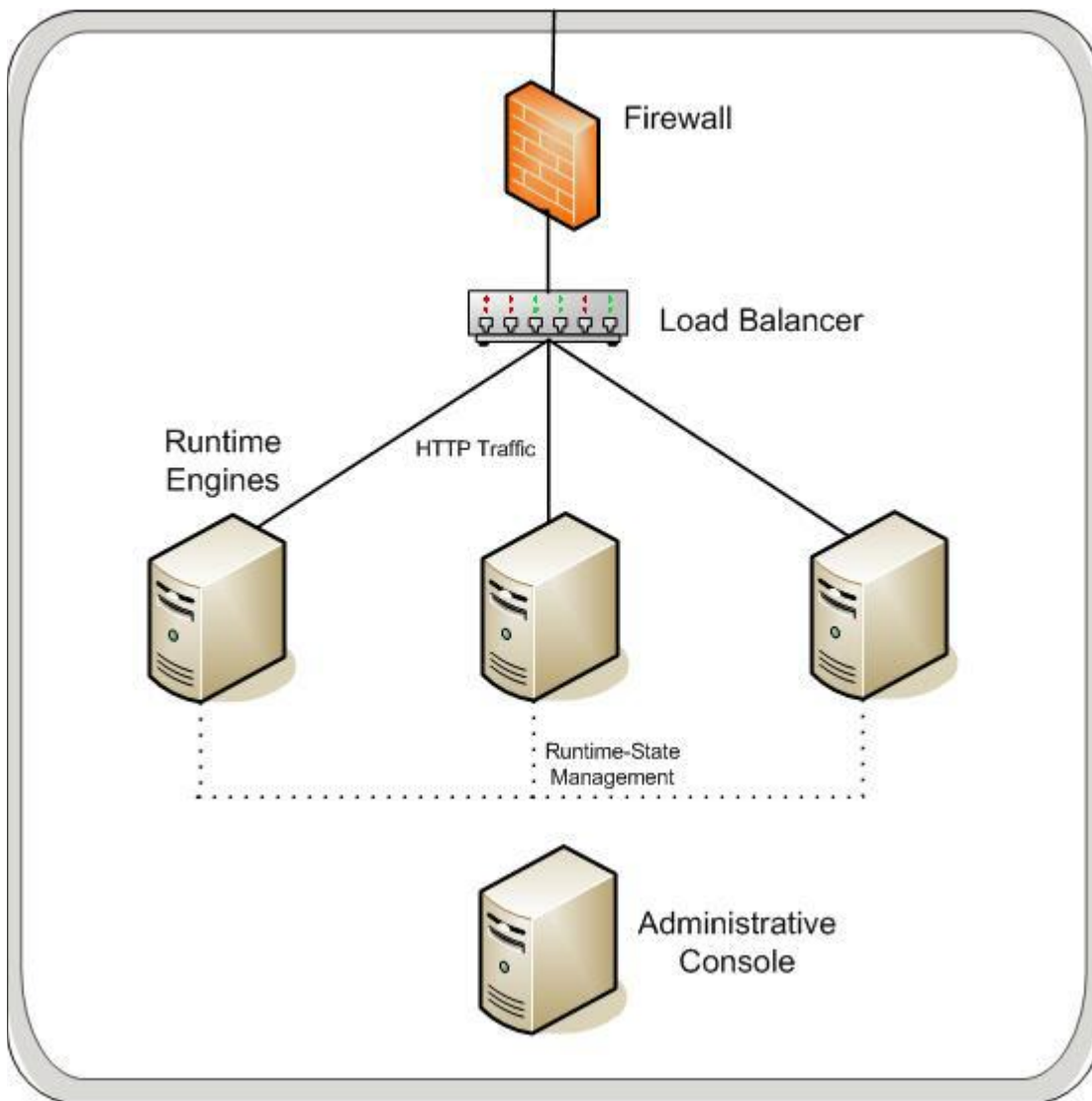
The use of preferred nodes can translate into any number of deployment configurations. Three primary strategies are discussed in the following sections and may serve as touch points for you to consider in conjunction with your network requirements:

- [Sharing All Nodes](#) (see next section)
- [Designating State Servers](#)
- [Defining Subclusters](#)

Tip: For PingFederate versions 7.0 and higher, it is possible to configure overrides for authentication-adapter processing based on the runtime node servicing a request (see *Configuring the Cluster Node Selector* in the PingFederate *Administrator's Manual*).

Sharing All Nodes

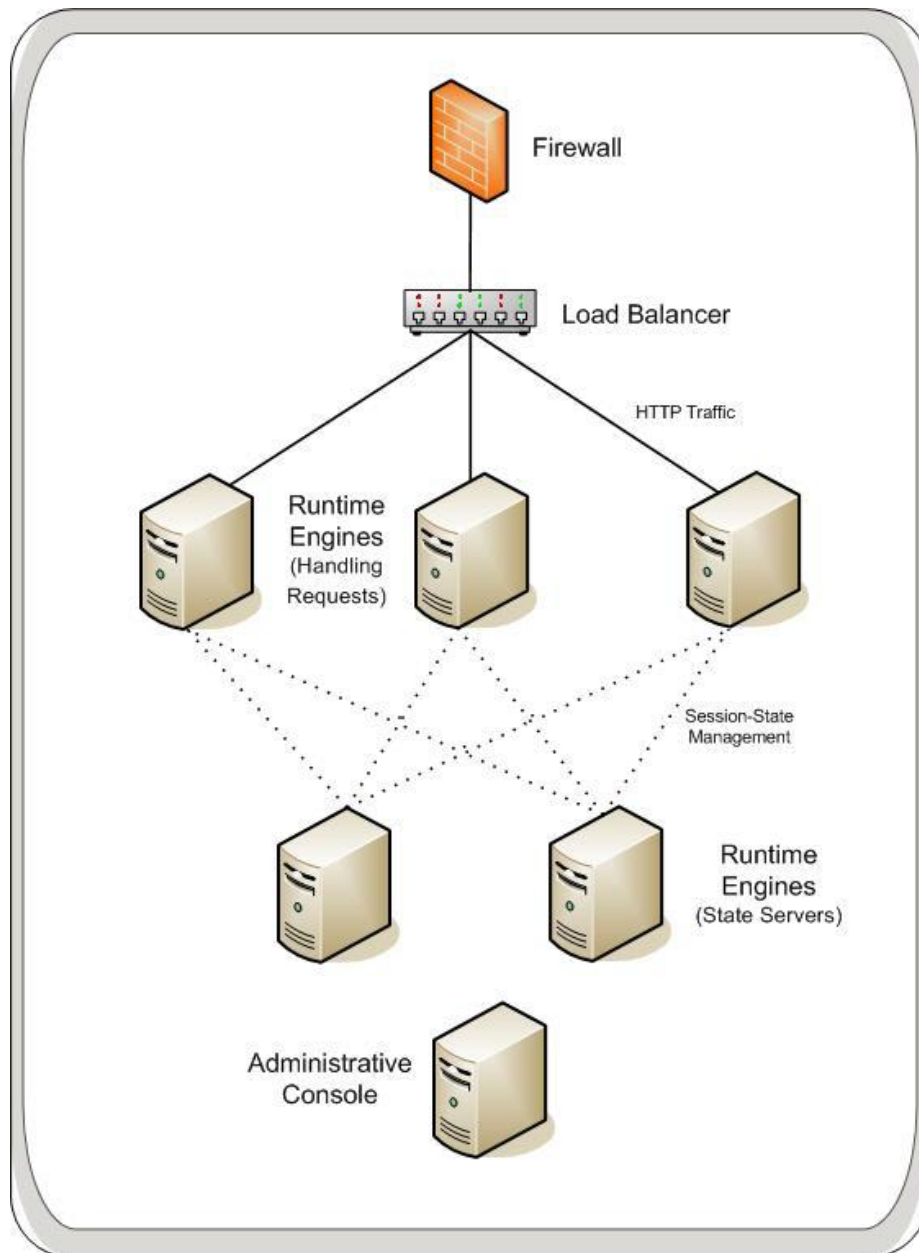
Leaving the preferred.node.indices property blank in all the cluster-configuration files provides a basic deployment case. An advantage of this approach is simplicity, including the option of using straightforward load-balancing strategies such as round robin. A disadvantage is that as additional nodes are added, the throughput improvement rate that clustering offers may decline as the state-replication overhead increases.



The diagram above illustrates the node-sharing approach. HTTP traffic is directed to all nodes.

Designating State Servers

You can select a few nodes as *state servers*. This deployment can be configured by setting the preferred-node indices of other servers in a group to those of the state servers. Load balancers should be configured to isolate the state-server nodes from end-user traffic. This approach scales better than the all-nodes approach because additional nodes do not require connections to every existing node; there need be only a connection between each server and each state server.

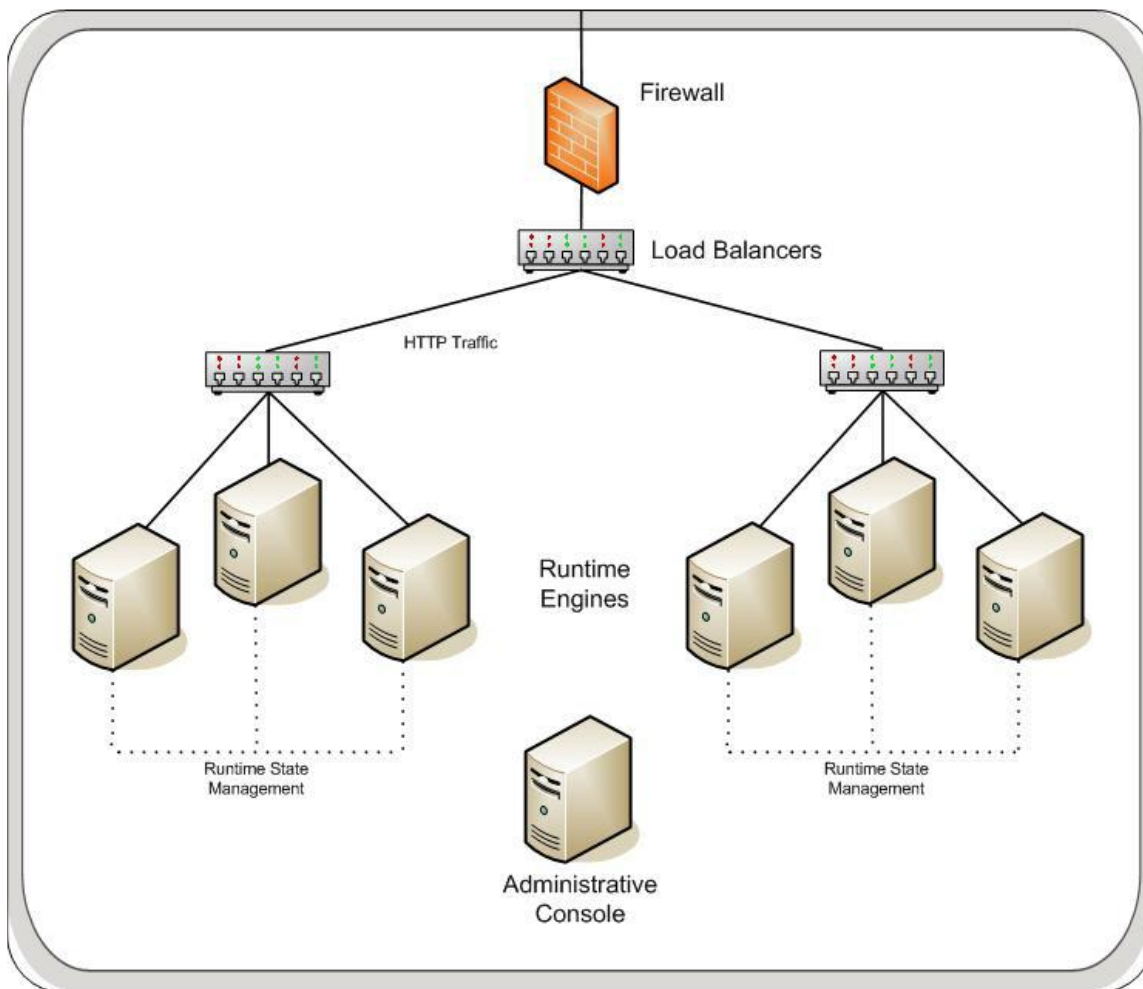


This diagram illustrates the state-server approach. If the two state-server nodes have indices of 1 and 2, then the preferred-nodes property of the other runtime nodes would be:

```
preferred.node.indices=1,2
```


Defining Subclusters

You can use node indices to divide a cluster into subgroups, or *subclusters*, of a few nodes each. Using this configuration, each node in a subcluster shares data only with other members of the subcluster. This approach requires the load balancer to persist, or *stick*, user sessions to the same group so that each subsequent request from the same user is directed to one of the nodes in that group. (A wide variety of persistence mechanisms exist and may vary by load balancer – consult the documentation specific to your product.) The advantage of this approach is that cluster throughput scales more linearly, because the creation of an additional subcluster will not degrade the performance of any other group.



The diagram above illustrates the subcluster approach. The preferred-nodes property of each server in the cluster lists the indices of all nodes in its subgroup (including itself). HTTP traffic is directed to all nodes, but the load balancer directs user sessions to the same subcluster.

Runtime State-Management Services

The Runtime State-Management Services are a collection of interfaces defining the contract that PingFederate uses with each service to manage session states. The implementation of each service interface is specified in the `hivemodule.xml` file in the `<pf_install>/pingfederate/server/default/conf/META-INF/` directory. This file does not need to be modified unless you wish to customize the way services are handled in a cluster. Any changes must be replicated manually for each cluster node.

By default, the interfaces listed in `hivemodule.xml` are proxies that select the best implementation for each service, based on the operational mode of the server. For example, if the server is in standalone mode, an in-memory-only implementation is used. If the server is in a clustered mode, a group RPC-based implementation is used. The proxies are provided for convenience; if you choose, you may specifically designate the implementation you desire for each service, as described in the following sections. Configuration files for the implementations that make use of preferred nodes are in the directory `<pf_install>/pingfederate/server/default/conf` (see [Managing Deployment Architecture](#) on page 8).

Inter-Request State Management Service

The PingFederate server tracks user-session state information between HTTP requests – for example, when PingFederate, acting as an Identity Provider (IdP), redirects a user’s browser to another system for authentication. When the user’s browser returns to PingFederate after authentication, the server needs access to the state associated with that user from before the redirection. Another example is keeping track of state between issuing a request to a partner and processing the response. Generally, this state is short-lived.

The `InterRequestStateMgmtProxy` interface chooses from among three methods available to track this state: group RPC-based (the clustering default), cookie based, and local memory-based.

Caution: Depending on deployments, if cookies are used for state management in a cluster instead of the default RPC-based method, keep in mind that the cookies may become large enough to exceed certain browser limitations.

Group RPC-based Session Tracking

This implementation uses the preferred-nodes concept; the configuration file is `cluster-inter-request-state.conf` in the `<pf_install>/pingfederate/server/default/conf/` directory. All three preferred-node approaches described under [Managing Deployment Architecture](#) on page 8 are possible with this implementation. The subgroup approach, however, may be most appropriate.

The service proxy `InterRequestStateMgmtProxy` in the `hivemodule.xml` file is set to use this implementation as the clustering default. The specific class name is:

```
org.sourceid.saml20.service.impl.grouprpc.InterRequestStateMgmtGroupRpcImpl
```

Cookie-Based Tracking

In a clustered mode, this implementation tracks short-lived session-state data with the help of browser cookies. The implementation works by compressing and encrypting the session-state data and sending it to the user's browser as a cookie. User data is then returned to the server on subsequent requests from the browser. No data for this service needs to be shared among nodes, and even the simplest round-robin load-balancing technique works. There is some overhead in the encryption and compression, but the advantage is that the session state must travel over the network only between the entities that actually need it – the user and the server that handles that user's next request.

Caution: Cookie-based tracking, which can result in larger cookies, *should not* be used for configurations using account linking. (For more information, see Account Linking in the *PingFederate Administrator's Manual*.)

This implementation does not use group RPC to share session-state data. However, RPC is used by default to handle dynamic distribution of the AES keys used for encryption and decryption (unless a static key is configured). The `key-tracker.xml` file, in the `<pf_install>/pingfederate/server/default/data/config-store` directory, controls the interval at which new keys are issued, the maximum number of keys to retain, and the key length. The oldest member of a cluster group is responsible for issuing and distributing new keys.

The file `inter-req-cookie-config.xml`, also in the `config-store` directory, specifies the name of the cookie, which you may modify as needed. You can also configure a static AES encryption key by specifying its hex value (the key must set on each node). While a static key is somewhat less secure, it eliminates the need for key management between nodes via back-channel communication.

To use this service implementation, set the class attribute for the `InterRequestStateMgmt` service in the `hivemodule.xml` file to the class name:

```
org.sourceid.saml20.service.impl.cookie.InterReqStateMgmtCookieImpl
```

Local Memory-based Tracking

In this alternative, the inter-request state of a user is tracked in the local memory of the processing server.

Note: To implement this alternative, the load balancer must support sticky sessions to force all requests for the same user session to be routed to the same server.

To use this service implementation, set the class attribute for the `InterRequestStateMgmt` service in the `hivemodule.xml` file to the class name:

```
org.sourceid.saml20.service.impl.localmemory.InterReqStateMgmtMapImpl
```

SP Session Registry Service

PingFederate uses the SP (Service Provider) Session Registry Service to facilitate single logout by tracking assertions issued from IdP partners. This service is used only when the PingFederate server is acting in an SP role and supports single logout with one or more partner connections.

When PingFederate is in clustered mode, the service proxy uses a group RPC-based, preferred-nodes implementation; the configuration file is `cluster-sp-session-registry.conf` in the `<pf_install>/pingfederate/server/default/conf` directory.

All preferred-node approaches are possible with this implementation (see [Managing Deployment Architecture](#) on page 8). However, if your site accepts single logout messages via SOAP, the subgroup approach may not succeed because the load balancer's sticky-session functionality cannot ensure that the SOAP message from a partner will be directed to the appropriate subgroup.

The service proxy uses the class:

```
org.sourceid.saml20.service.impl.grouprpc.SpSessionRegistryGroupRpcImpl
```

IdP Session Registry Service

PingFederate uses the IdP Session Registry Service to facilitate single logout by tracking assertions issued to SP partners. This service is used only when the PingFederate server is acting in an IdP role and supports single logout with one or more partner connections.

When PingFederate is in clustered mode, the service proxy uses a group RPC-based, preferred-nodes implementation; the configuration file is `cluster-idp-session-registry.conf` in the `<pf_install>/pingfederate/server/default/conf` directory.

As with the SP service, all three preferred-node approaches are possible with this implementation (see [Managing Deployment Architecture](#) on page 8), but inbound SOAP logout messages may not be handled correctly for subgroups.

The service proxy uses the class:

```
org.sourceid.saml20.service.impl.grouprpc.IdpSessionRegistryGroupRpcImpl
```

LRU Memory Management Schemes

During the normal course of transaction processing, the SP and IdP Session Registries and the Inter-Request State Management service manage memory as part of normal processing. However, it is common for end users to abandon Web sessions, resulting in orphaned data. To ensure that such data does not result in excessive memory usage, the data structures used by these services employ a least-recently-used (LRU) algorithm to purge old data. When a data structure reaches the maximum size, the oldest entries are automatically removed.

The maximum size of each data structure is configurable in the file `size-limits.conf` in the `<pf_install>/pingfederate/server/default/conf` directory.

Assertion Replay Prevention Service

The SAML (Security Assertion Markup Language) standards specify that when an SP receives assertions via the POST binding, the SP should keep track of that assertion for the duration of its validity to ensure that it is not replayed (that is, intercepted by a third party and reposted). PingFederate delegates that responsibility to the Assertion-Replay Prevention Service. This service is used only when PingFederate is acting as an SP and receives assertions via the POST binding (see Supported Standards in *Getting Started* for information about SAML bindings).

When PingFederate is in clustered mode, the service proxy uses a group RPC-based, preferred-nodes implementation; the configuration file is `cluster-assertion-replay-prevention.conf` in the `<pf_install>/pingfederate/server/default/conf` directory.

Due to the nature of the threat that this service is intended to prevent, only the [Sharing All Nodes](#) and [Designating State Servers](#) deployment strategies are appropriate for this service (see after [Managing Deployment Architecture](#) starting on page 8).

The service proxy uses the class:

```
org.sourceid.saml20.service.impl.grouprpc.  
AssertionReplayPreventionServiceGroupRpcImpl
```

This service is unique in that it fulfills only a security condition of the federation specifications, rather than supporting normal SSO functionality. Because many other security requirements are in place (for example, SSL, no-cache directives, and digital signatures), there may be situations where the priority placed on cluster performance outweighs the priority placed on this security check. If you are in this situation, you may wish to change the implementation for the service point `AssertionReplayPreventionService` in the `hivemodule.xml` file to one of these classes:

- `org.sourceid.saml20.service.impl.localmemory.
AssertionReplayPreventionSvcInMemoryImpl`

This is the implementation used in standalone mode. It performs all the appropriate replay checks but does not share any data with other nodes. A replay attempt routed to the same server node would fail, but other nodes would not have sufficient information to stop the transaction.

- `org.sourceid.saml20.service.impl.localmemory.
AssertionReplayPreventionServiceNullImpl`

This implementation disables assertion-replay prevention; however, you may wish to use it, with caution, when performance is an absolute priority.

Artifact-Message Persistence and Retrieval Service

When the artifact binding is used to send messages to partners, the PingFederate server must keep track of the message referenced by the artifact until the partner makes a SOAP call to retrieve it.

Two options are available for using outbound artifacts in a cluster:

- Group RPC-Based Retrieval
- Using SAML 2.0 Indexing

Group RPC-Based Retrieval

When PingFederate is in clustered mode, the service proxy selects a group RPC-based implementation, which takes advantage of node indexing but does not use the preferred-nodes concept. Sticky-session load-balancing strategies are not effective for the artifact binding, because the requests that result in the issuance of the artifact and the artifact resolution request come from different locations.

This implementation works by having the node that issues an artifact encode its node index into the artifact using two bytes of the artifact message handle. When a server receives the artifact resolution

request, it uses the encoded node index to determine the issuing node and makes an RPC call to get the associated message.

Although this implementation does not use the preferred-nodes concept, it does have a configuration file, `<pf_install>/pingfederate/server/default/conf/cluster-artifact.conf`, in which the RPC time-out can be configured.

The service proxy uses the class:

```
org.sourceid.saml20.service.impl.grouprpc.  
ArtifactPersistenceSvcGroupRpcEncodedNodeIdxImpl
```

SAML 2.0 Indexing (Local Memory)

Due to the implementation complexity involved in managing state with the artifact binding, the SAML 2.0 specification introduced the concept of indexed endpoints. A SAML 2.0 federation entity may support multiple artifact resolution services, each identified by a unique index number. Artifacts include this index, and a federation partner must send the artifact resolution request to the appropriate endpoint for that index. This means that servers do not need to share information concerning the artifact.

The downside to this approach is that partners must know about each of your back-end servers. Generally, this means providing partners with a list that includes multiple artifact-resolution service endpoints with the corresponding indices.

Note: PingFederate does not automatically generate this information; an administrator must create it and send it to partners who are using the artifact binding.

For example, if you have four servers in a cluster, the list might look like this:

```
<ArtifactResolutionService Binding="..."  
Location="https://node1/idp/ARS.ssaml2" index="1"/>  
<ArtifactResolutionService Binding="..."  
Location="https://node2/idp/ARS.ssaml2" index="2"/>  
<ArtifactResolutionService Binding="..."  
Location="https://10.0.1.72/idp/ARS.ssaml2" index="3"/>  
<ArtifactResolutionService Binding="..."  
Location="https://node4/idp/ARS.ssaml2" index="4"/>
```

In this case, the index corresponds to the node index configured in the `run.properties` file on each individual server. This service encodes the node index in the artifact handle when running in a clustered mode (it will always use an index of zero in standalone mode).

Not only do partners need to know about each back-end server, they need direct access to each ARS endpoint. This may require more complicated configuration of load balancers, proxies, and firewalls. Another drawback to this approach is that it cannot be used for SAML 1.x, or with adapters that utilize PingFederate's artifact-data management.

To use this approach for SAML 2.0 federation deployments, edit the `hivemodule.xml` file and change the implementation for the `ArtifactStore` service point to the class name:

```
org.sourceid.saml20.service.impl.localmemory.  
ArtifactPersistenceServiceMapImpl
```

Back-Channel Session Revocation Service

PingFederate uses the Back-Channel Session Revocation Service to provide OAuth clients the capabilities to add sessions to the revocation list and to query the revocation status (see Session Revocation API in the *Administrator's Manual*).

When PingFederate is in clustered mode, the service proxy uses a group RPC-based implementation. When adding a session to its revocation list, the processing node always propagates the information to all engine nodes in the cluster. It does not use the preferred-nodes concept. This enables the flexibility of allowing the queries to be processed locally or results to be returned after collecting the information from other engine nodes; the former yields faster response time for engine nodes that are deployed in well-connected networks while the latter adds a layer of protection against inconsistent revocation lists among the engine nodes due to possible network outages.

The configuration file is `<pf_install>/pingfederate/server/default/conf/cluster-session-revocation.conf`. This is where the RPC time-out and other settings can be tuned (see [Managing Deployment Architecture](#) on page 8).

The service proxy uses the class:

```
org.sourceid.saml20.service.impl.grouprpc.  
SessionRevocationServiceGroupRpcImpl
```

FIFO Memory Management Scheme

To ensure the revocation list does not result in excessive memory usage, in addition to the Session Revocation Lifetime setting (see Authorization Server Settings in the *Administrator's Manual*), the back-channel session revocation service employs a first-in-first-out (FIFO) algorithm to purge old data. When the maximum size is reached, the oldest entries are automatically removed.

The maximum number of sessions is configurable by the `SessionRevocationServiceMapImpl.max.revoked.sris` setting in `<pf_install>/pingfederate/server/default/conf/size-limits.conf`. The default value is 50000.

Extensibility and Other Services

Custom implementations of all state management services described in this *Guide* can be plugged into PingFederate as needed for special deployments. Software developers can refer to the Javadoc reference:

```
<pf_install>/pingfederate/sdk/doc/index.html
```

Two other services may need consideration when running PingFederate in a cluster, depending on SAML 2.0 federation deployment needs:

Account Linking Service

This service stores the association between the external and internal identifiers of an end user when account linking is used as an SP identity-mapping strategy (see Key Concepts in the *Administrator's Manual* for more information). The default, standalone implementation uses a JDBC interface to an embedded database within PingFederate. No information from the embedded database is shared across the cluster. Therefore, when account linking is used for an IdP connection deployed in a cluster, the

default implementation will not work properly. In such cases, the pointer must be adjusted for cluster use by pointing the service to an external database (see *Defining an Account-Linking Data Store* in the *PingFederate Administrator's Manual*).

Pseudonym Service

This service references the method needed by PingFederate to generate or look up a pseudonym for a user (see *Key Concepts* in the *Administrator's Manual* for more information). The service is used only if your site is acting in an IdP role and produces assertions containing pseudonyms as subject identifiers. The default implementation uses a message digest to produce the value so that no session-state synchronization is required. However, it may be desirable in some situations to implement pseudonym handling differently. Developers can refer to the Javadoc reference describing `PseudonymService` interface for more information.

Deploying Provisioning Failover

If you are using PingFederate as an Identity Provider (IdP) and have enabled and configured Outbound Provisioning, then you can set up one or more failover PingFederate servers specifically for provisioning backup.

Provisioning runtime processing and failover is independent of SSO/SLO runtime processing and server clustering described in the preceding sections. However, if you are already deploying, or have deployed, a cluster for federation-protocol runtime processing, then you can use a subset of those servers for provisioning failover. Alternatively, you can mix the configuration or set up provisioning-failover servers independently.

Important: Hypersonic databases, including the default database bundled with PingFederate, cannot be used in a failover configuration. Each server in the failover network must be configured to use the same relational database (for example, Oracle). (For more information, see *Key Concepts* in the *PingFederate Administrator's Manual*.)

To deploy failover for Outbound Provisioning:

1. Identify two or more runtime instances of PingFederate to configure for provisioning failover.
Install new instances of PingFederate if needed (see *Installation* in *Getting Started*).
2. For each server instance, edit provisioning properties in the `run.properties` file located in the `<pf_install>/pingfederate/bin` directory (see the next section, [Configuring Runtime Properties](#)).
3. Start or restart all of the PingFederate servers.
Refer to *Starting and Stopping PingFederate*, in the *PingFederate Administrator's Manual*, if needed.
4. After you configure (or reconfigure) Outbound Provisioning in the administrative console, replicate the configuration on each server (see [Synchronizing Server Runtime Configuration](#) on page 20).

The *Administrator's Manual* and installed **Help** provide detailed information on using the administrative console to configure provisioning.

Configuring Runtime Properties

The `run.properties` in the `<pf_install>/pingfederate/bin` directory contains runtime failover settings. Configure the properties described in the following table for each server in the failover network.

Property	Description
<code>pf.provisioner.mode</code>	Enables or disables provisioning. Allowed values are: OFF – Outbound Provisioning is disabled (the default). STANDALONE – Provisioning is enabled, without failover. FAILOVER – Provisioning is enabled, with failover. Important: The STANDALONE setting is not used for failover configuration; primary and secondary server(s) must be set to FAILOVER.
<code>provisioner.node.id</code>	Each server must have a unique index number (from 1 to <i>n</i>), which is used to prioritize which server is currently active and which is next in line in case of a failure. Important: The active primary server <i>must</i> have an index number of 1.
<code>provisioner.failover.grace.period</code>	The time interval (in seconds) between the first indication that a node is dead and failover to the next server in line. The time period should be greater than the Synchronization Frequency set in the administrative console (see Configuring Outbound Provisioning Settings the PingFederate <i>Administrator's Manual</i>).

Synchronizing Server Runtime Configuration

All nodes in a cluster must have the same configuration settings (as set via the administrative console). This information includes such settings as partner-connection endpoints, data stores, and certificates, as well as Outbound Provisioning configuration data, when applicable.

For a server cluster, you can use either of the following methods to ensure that configuration data is synchronized on all cluster nodes:

- Console Configuration Push
- Configuration-Archive Deployment

Tip: PingFederate also provides a Web Service for synchronizing clustered servers, as part of the Connection Management Service. For information, refer to Web Service Interfaces in the PingFederate *Administrator's Manual*.

For exclusive or mixed-use Outbound Provisioning failover, when the primary and/or backup server(s) are not otherwise part of a cluster, you must use the configuration-archive method (see the previous section).

Note: Changes made directly to configuration files (for example, in `<pf_install>/pingfederate/bin/run.properties`) must be replicated manually across the cluster, as applicable, and PingFederate servers must be restarted if running.

Console Configuration Push

When running in cluster mode, the administrative console provides a **Cluster Management** link on the Main Menu under Administrative Functions. Whenever applicable changes are made to the administrative-console configuration, a message appears at top of the console directing the administrator to this screen.

ADDRESS	INDEX
172.17.17.163:7600 (Administrative console)	1
172.17.18.197:7600	2
172.17.17.12:7600	3

Replicate Configuration Refresh Table

Last Modified: Tue Mar 19 13:19:06 GMT-07:00 2013
Last Replicated: Tue Mar 19 13:19:24 GMT-07:00 2013

From this screen, replicate the current console configuration to all the server nodes in the cluster if the Last Modified timestamp is later than the Last Replicated timestamp.

Tip: The replication process takes only a moment during which time any new requests coming into any of the cluster nodes are held in a queue. When the replication is complete, processing continues where it left off.

Click Refresh Table to verifying that any new nodes have joined the cluster.

Configuration-Archive Deployment

After you configure or reconfigure the console, you can also update cluster nodes by downloading a configuration archive and then deploying it (either manually or via a scripted process) to each cluster node or provisioning-failover server. For more information about creating and deploying configuration archives, see System Administration in the *Administrator's Manual*.

A configuration archive contains the same information sent during the configuration push from the administrative console described in the previous section. However, this option provides for scheduling and scripting cluster synchronization.

Note: You must still configure each node's session-state services, if you are using settings other than defaults (see [Managing Deployment Architecture](#) on page 8). The archive does not contain information about clustering; it contains only information about server settings and partner configuration.

PingFederate®

SSO Integration Overview



© 2006-2015 Ping Identity® Corporation. All rights reserved.

PingFederate SSO *Integration Overview*
Version 7.3
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909
Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: **January 28, 2015**.

Contents

- Integration Introduction4**
- SSO Integration Concepts4**
- Identity Provider Integration5**
 - Custom Applications6
 - Identity Management Systems6
 - Authentication Systems.....7
- Service Provider Integration8**
 - Custom Applications8
 - Server Agents9
 - Identity Management Systems9
 - Commercial Applications.....10
- Summary10**

Integration Introduction

As a stand-alone server, PingFederate must be integrated programmatically with end-user applications and identity management (IdM) systems to complete the “first- and last-mile” implementation of a federated-identity network. The purpose of this document is to provide an overview of the various approaches to integrating systems and applications with PingFederate for browser-based single sign-on (SSO). To enable both the Identity Provider (IdP) and Service Provider (SP) sides of this integration, PingFederate provides commercial integration kits, which include *adapters* that plug into the PingFederate server and *agents* that interface with local IdM systems or applications.

Tip: Integration kits are available for download at pingidentity.com (pingidentity.com/support-and-downloads). Find *User Guides* and other documentation for the kits under [Product Documentation](https://pingidentity.com/support-and-downloads) on the Web site (documentation.pingidentity.com/display/LP/Product+Documentation).

This document covers the integration kits available from Ping Identity for PingFederate. PingFederate also includes a robust software development kit (SDK), which software developers can use to write their own custom interfaces for specific systems. Please refer to the PingFederate SDK Developer's Guide for more information, available in the PingFederate distribution `sdk` directory.

Note: Ping Identity offers separate integration solutions for secure SSO to Software-as-a-Service (SaaS) providers—SaaS Connectors, which include automatic user provisioning at the provider site. In addition, for integration with the PingFederate WS-Trust Security Token Service (STS), we provide a range of *Token Translators*. These plug-in Token Processors (for an IdP) and Generators (for an SP) connect the STS with Web Service Providers and Clients for access to identity-enabled Web Services.

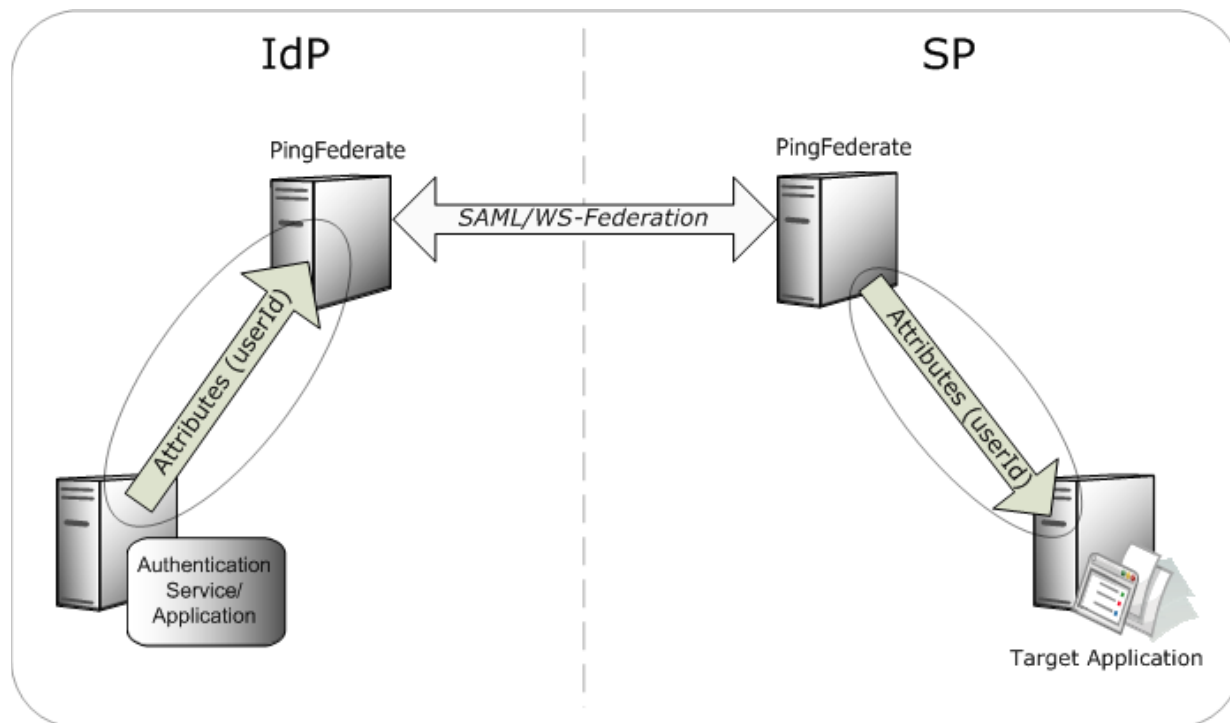
For more information about SaaS Connectors and Token Translators, refer to Key Concepts in the PingFederate *Administrator's Manual*. For lists of available Connectors and Translators, go to [Support and Downloads](https://pingidentity.com/support-and-downloads) on the Ping Identity Web site (pingidentity.com/support-and-downloads).

SSO Integration Concepts

For an IdP, the first step in the integration process involves sending identity attributes from an authentication service or application to PingFederate. PingFederate uses those identity attributes to generate a SAML assertion. (For information about SAML—Security Assertion Markup Language—refer to Supported Standards in *Getting Started*.) IdP integration typically provides 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 step of the integration process involves sending identity attributes from PingFederate to the target application. PingFederate extracts the identity attributes from the incoming SAML assertion and sends them to the target application to set a valid session cookie or other application-specific security context for the user.

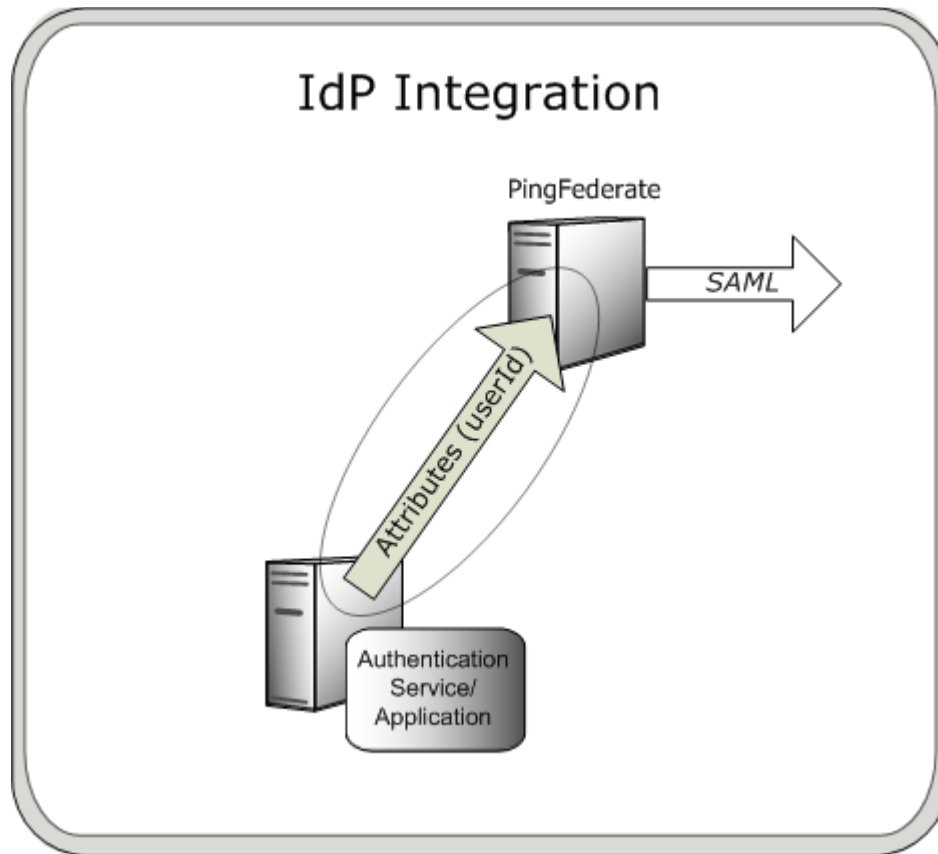
The following diagram illustrates the basic concepts of integration with PingFederate:



Identity Provider Integration

An IdP is a system entity that authenticates a user, or “SAML subject,” and transmits referential identity attributes based on that authentication to PingFederate. The IdP integration involves retrieving user-identity attributes from the IdP domain and sending them to the PingFederate server. Typically, the identity attributes are retrieved from an authenticated user session. For IdP integration, a number of attribute-retrieval approaches can be used, depending upon the IdP deployment/implementation environment. Ping Identity offers a broad range of commercial integration kits that address various IdP scenarios, most of which involve either custom-application integration, integration with a commercial IdM product, or integration with an authentication system.

Tip: For IdPs implementing SSO to selected Software-as-a-Service (SaaS) providers—for example, Google Apps and Salesforce—PingFederate also provides for automated user provisioning. See details under [Workforce to the Cloud](#) at the Ping Identity Web site.



Custom Applications

A federation partner can use a custom authentication service or application to serve as the IdP role in that federation partnership. Integration with a custom application is handled through application-level integration kits, which allow software developers to integrate their custom applications with a PingFederate server acting as an IdP. Each application-level integration kit includes an agent, which resides with the IdP application and provides a simple programming interface to transfer session and attribute information from the application to the PingFederate IdP server.

Ping Identity provides custom-application integration kits for several programming environments, including:

- Java
- .NET
- PHP

In addition, Ping Identity provides an Agentless Integration Kit, which allows developers to use direct HTTP calls to the PingFederate server to temporarily store and retrieve user attributes securely, eliminating the need for an agent interface.

Identity Management Systems

An IdP enterprise that uses an IdM system can expand the reach of the IdM domain to external partner applications through integration with PingFederate. IdM integration kits typically use the IdM agent

API (if available) to access identity attributes in the IdM proprietary session cookie and transmit those attributes to the PingFederate server.

IdM integration kits do not require any development; integration with PingFederate is accomplished entirely through the PingFederate administrative console.

Ping Identity provides integration kits for many of the leading IdM systems including:

- Oracle Access Manager (COREid)

Authentication Systems

Initial user authentication is normally handled outside of the PingFederate server using an authentication application or service. PingFederate authentication-system integration kits leverage this local authentication to access applications outside the security domain. For example, an integration kit might access authenticated sessions that are validated against a Windows NTLM or Integrated Windows Authentication (IWA) environment, and then pass session attributes to the PingFederate IdP server.

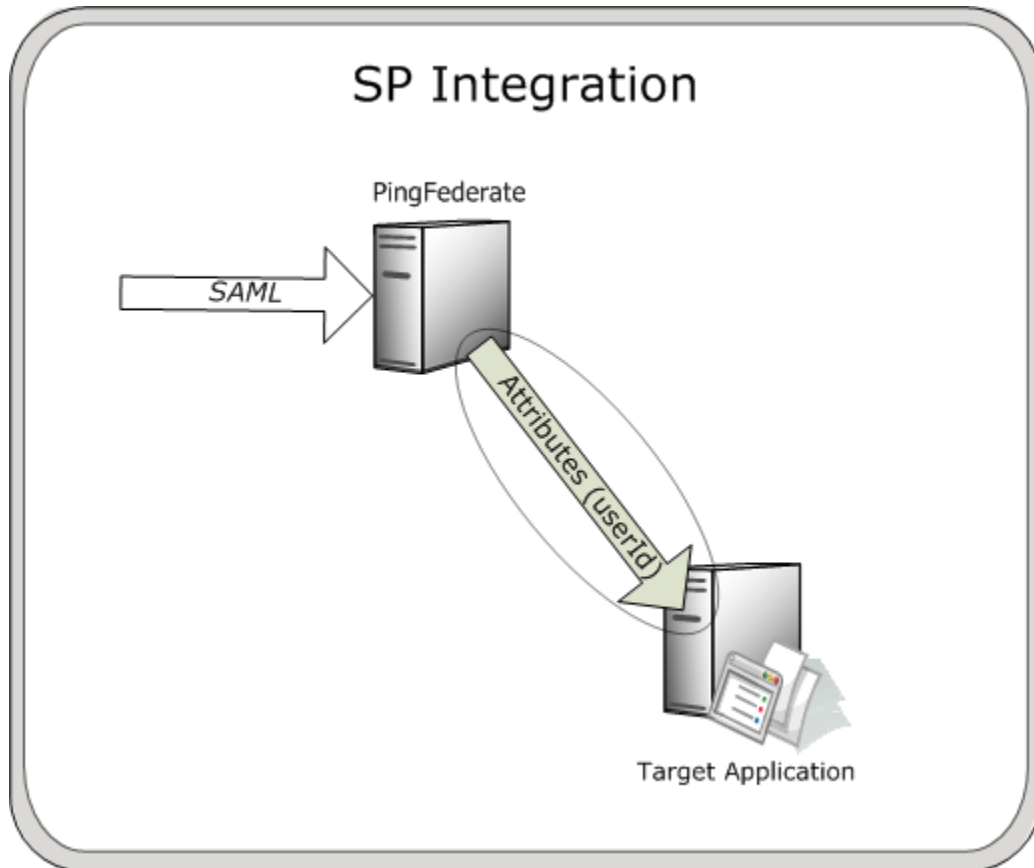
Authentication integration kits do not require any development; integration with PingFederate is accomplished entirely through the PingFederate administrative console. Ping Identity offers integration kits for authentication systems including:

- IWA/NTLM
- X.509 Certificate
- RSA SecurID® Integration Kit
- VeriSign Identity Protection Integration Kit

PingFederate also packages two IdP adapters, an HTML Form Adapter and an HTTP Basic Adapter, which delegate user authentication to plug-in Password Credential Validators. Supplied Validators can use either an LDAP directory, RADIUS server, or a simple username/password verification system maintained by PingFederate. (Customized Validators may also be developed.) When the PingFederate IdP server receives an authentication request for SP-initiated SSO or a user clicks a link for IdP-initiated SSO, PingFederate invokes the implemented adapter and prompts the user for credentials, if the user is not already logged on.

Service Provider Integration

An SP is the consumer of identity attributes provided by the IdP through a SAML assertion. SP integration involves passing the identity attributes from PingFederate to the target SP application. The SP application uses this information to set a valid session or other security context for the user represented by the identity attributes. Session creation can involve a number of approaches, and as for the IdP, Ping Identity offers commercial integration kits that address the various SP scenarios. Most SP scenarios involve custom-application integration, server-agent integration, integration with an IdM product, or integration with a commercial application.



Custom Applications

Many applications use their own authentication mechanisms, typically through a database or LDAP repository, and are responsible for their own user-session management. Custom-application integration is necessary when there is limited or no access to the Web or application server hosting the application. Integration with these custom applications is handled through application-level integration kits, which allow software developers to integrate their applications with a PingFederate server acting as an SP.

With these integration kits, PingFederate sends the identity attributes from the SAML assertion to the SP application, which can then use them for its own authentication and session management. As for the IdP, application-specific integration kits include an SP agent, which resides with the SP application and provides a simple programming interface to extract the identity attributes sent from the PingFederate server. The information can be used to start a session for the SP application.

Ping Identity provides custom-application integration kits for a variety of programming environments, including:

- Java
- .NET
- PHP

In addition, Ping Identity provides an Agentless Integration Kit, which allows developers to use direct HTTP calls to the PingFederate server to temporarily store and retrieve user attributes securely, eliminating the need for an agent interface.

Server Agents

Server-agent integration with PingFederate allows SP enterprises to accept SAML assertions and provide SSO to all applications running on that Web and/or application server; there is no need to integrate each application. Since integration occurs at the server level, ease of deployment and scalability are maximized. Applications running on the Web/application server must delegate authentication to the server; if the application employs its own authentication mechanism, integration must occur at the application level.

With server-agent integration kits, PingFederate sends the identity attributes from the SAML assertion to the server agent, which is typically a Web filter or JAAS Login Module. The server agent extracts the identity attributes, which the server then uses to authenticate and create a session for the user.

SP server-agent integration kits do not require any development; integration with PingFederate is accomplished entirely through the PingFederate administrative console.

Ping Identity provides integration kits for many Web and application servers, including:

- Internet Information Services (IIS)
- Apache (Red Hat)
- Apache (Windows)
- WebSphere
- SAP NetWeaver®

Identity Management Systems

IdM integration with PingFederate allows an SP enterprise to accept SAML assertions and provide SSO to applications protected by the IdM domain. IdM integration kits typically use the IdM agent API (if available) to create an IdM proprietary session token based on the identity attributes received from PingFederate.

IdM integration kits do not require any development; integration with PingFederate is accomplished through the PingFederate administrative console and the IdM administration tool.

Ping Identity provides integration kits for leading IdM systems including:

- Oracle Access Manager (formerly COREid)

Commercial Applications

Commercial-application integration with PingFederate allows an SP enterprise to accept SAML assertions and provide SSO to those commercial applications.

These integration kits do not require any development; integration with PingFederate is accomplished entirely through the PingFederate administrative console.

Ping Identity offers integration kits for these commercial applications:

- Citrix
- SharePoint
- Salesforce.com

Note: For PingFederate 5.2 and later versions, the Salesforce.com Integration Kit is called the PingFederate Salesforce *Connector*. Connectors feature complete user provisioning for SaaS providers, as well as SSO quick-connection templates.

Summary

The following table summarizes IdP- and SP-integration deployment scenarios and the Ping Identity bundled adapters and integration kits that suit each scenario. Ping Identity continues to develop new bundled adapters (included with PingFederate) and integration kits; check the Ping Identity [download site](http://www.pingidentity.com/support-and-downloads) (www.pingidentity.com/support-and-downloads) for the most up-to-date list of kits.

Please find *User Guides* and other documentation for current integration kits under [Product Documentation](#) on the Web site (documentation.pingidentity.com/display/LP/Product+Documentation).

Type	IdP	SP
Custom Application	Java Integration Kit .NET Integration Kit PHP Integration Kit Agentless Integration Kit	Java Integration Kit .NET Integration Kit PHP Integration Kit Agentless Integration Kit
Identity Management System (IdM)	Web Access Management (WAM) Integration Kit	WAM Integration Kit

Type	IdP	SP
Authentication System	Windows IWA/NTLM Integration Kit X.509 Certificate Integration Kit HTML Form Adapter (Bundled with PingFederate) HTTP Basic Adapter (Bundled with PingFederate) RSA SecurID Integration Kit VeriSign Identity Protection Integration Kit	N/A
Server Agent	Integration Kit for SAP NetWeaver	IIS Integration Kit Apache Integration Kit (Red Hat) Apache Integration Kit (Windows) WebSphere Integration Kit Integration Kit for SAP NetWeaver
Commercial Application	N/A	Salesforce Connector Citrix Integration Kit SharePoint Integration Kit

PingFederate[®]

SDK Developer's Guide



© 2005-2015 Ping Identity® Corporation. All rights reserved.

PingFederate SDK *Developer's Guide*
Version 7.3
February, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)

Fax: 303.468.2909

Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: February 4, 2015.

Contents

Preface	4
Intended Audience	4
Additional Documentation	4
SDK Introduction	4
Adapter and STS Token-Translator Interfaces.....	5
Authentication Selector Interfaces	5
Custom Data Source Interfaces.....	5
Password Credential Validator Interfaces	6
Identity Store Provisioner Interfaces.....	6
Ping Identity Global Client Services.....	6
Getting Started With the SDK	6
Directory Structure	6
Setting Up Your Project.....	7
Implementation Guidelines	7
Shared Interfaces.....	8
IdP Adapter Implementation	9
SP Adapter Implementation.....	11
Token Processor Implementation	13
Token Generator Implementation	14
Authentication Selector Implementation.....	14
Custom Data Source Implementation	15
Password Credential Validator Implementation.....	17
Identity Store Provisioner Implementation.....	18
Building and Deploying Your Project	23
Building and Deploying With Ant	23
Manually Building and Deploying.....	25
Logging.....	25

Preface

This document provides technical guidance for using the Java Software Development Kit (SDK) for PingFederate. Developers can use this *Guide*, in conjunction with the installed Javadocs, to extend the functionality of the PingFederate server.

Intended Audience

The *Guide* is intended for application developers and system administrators responsible for extending PingFederate, including development of:

- Authentication adapters needed to integrate Web applications or identity-management systems (when not already available: see the PingFederate [SSO Integration Overview](#), described under [Additional Documentation](#), below)
- Authentication Selectors used to direct SSO authentication to instances of authentication sources based on specified conditions
- WS-Trust Security Token Service (STS) token translators, including token processors needed to consume and validate security tokens and token generators needed to create security tokens
- Custom data source drivers
- Password credential validators
- Identity store provisioners

The reader should be familiar with Java software-development principles and practices.

Additional Documentation

- Javadocs provide detailed reference information for developers. The Javadocs are located in the `<PF_install>/pingfederate/sdk/doc` directory.
- The PingFederate [SSO Integration Overview](#) describes the types of prebuilt authentication adapters Ping Identity provides for integrating Web applications and identity-management systems with PingFederate. Since these adapters are based on the SDK, you may want to review this document before building your own adapter to see if your needs have already been met.
- The PingFederate [Administrator's Manual](#) provides background information and user-interface (UI) configuration details needed to integrate implementation(s) of PingFederate interfaces.
- Integration Kit User Guides for [Java](#), [.NET](#), and [PHP](#) show examples of SDK implementations.

SDK Introduction

The PingFederate Java SDK consists of several Application Programming Interfaces (APIs), including:

- Adapter and STS Token-Translator Interfaces
- Authentication Selector Interfaces

- Custom Data Source Interfaces
- Password Credential Validator Interfaces
- Identity Store Provisioner Interfaces

Each of these interfaces allows users to create their own plug-ins, customizing certain behaviors of PingFederate to suit an organization's needs. This SDK provides a means to develop, compile, and deploy custom plug-ins to PingFederate.

A number of example plug-ins are included in the PingFederate package for reference. The example projects are located in the <PF_install>/sdk/plugin-src directory.

Adapter and STS Token-Translator Interfaces

The adapter and token-translator APIs enable PingFederate integration with IdPs or SPs. The APIs allow developers to build their own custom implementations for communicating authentication and security information between PingFederate and the enterprise environment.

Note: Token-translator interfaces are applicable only to PingFederate versions 6.0 and higher.

In addition to providing requisite runtime integration, an adapter or token translator also describes its configuration parameters to PingFederate; this enables the administrative console to render configuration screens with extensible validation.

Note: Suitable adapter or token-translator implementations for your deployment may already exist, or new implementations may be under development. Before developing your own custom solution, see the [Downloads](http://www.pingidentity.com/support-and-downloads) page (www.pingidentity.com/support-and-downloads) for more information about currently available implementations.

Authentication Selector Interfaces

Authentication selectors provide a mechanism to choose among multiple authentication sources and to direct a user to use a particular adapter or IdP connection (for federation hub use cases), depending on the specified conditions. For example, an authentication selector may map internal corporate users to use one adapter, while it maps external non-corporate users to a different adapter.

Note: Authentication selector interfaces are applicable only to PingFederate versions 7.3 and higher.

Authentication selectors are configurable UI plug-ins, allowing you to render custom configuration screens.

Custom Data Source Interfaces

The custom data source API is a set of Java interfaces that enable PingFederate to integrate with data stores not covered by existing LDAP or JDBC drivers. This allows developers to retrieve attributes from a data source of their choice during attribute fulfillment for various use cases. Similar to the adapter API, custom data source plug-ins also provide much of the same UI configuration functionality.

Password Credential Validator Interfaces

The password credential validator interfaces allow developers to define credential validators that are used to verify a given username and password in various contexts throughout the system. For example, credential validators are used to configure OAuth Resource Owner authorization grants and the HTML Form IdP Adapter.

Note: Credential validator interfaces are applicable only to PingFederate versions 6.5 and higher.

Identity Store Provisioner Interfaces

Identity Store Provisioners provide a mechanism for provisioning and deprovisioning users and (optionally) groups to external user stores. For example, a custom Identity Store Provisioner could be configured within an Inbound Provisioning IdP Connection to provision users using the SCIM protocol.

Note: Identity Store Provisioner interfaces are applicable only to PingFederate versions 7.1 and higher.

Similar to the adapter API, Identity Store Provisioners are configurable UI plug-ins, allowing you to render custom configuration screens.

Ping Identity Global Client Services

If you need assistance in using the SDK, visit the Ping Identity [Support Center](http://www.pingidentity.com/support) (www.pingidentity.com/support) to see how we can help you with your application.

Getting Started With the SDK

This section describes the directories and build components that comprise the SDK and provides instructions for setting up a development environment.

Directory Structure

The PingFederate SDK directory (`<PF_install>/pingfederate/sdk`) contains the following:

- `plugin-src/` – The directory where you place your custom plug-in projects. This directory also contains example plug-in implementations showing a wide range of functionality. You may use these examples for developing your own implementations.
- `doc/` – Contains the SDK Javadocs. Open `index.html` to get started.
- `lib/` – Contains libraries used for compiling and deploying custom components into PingFederate.
- `build.properties` – This file contains properties used by the Ant build script, `build.xml`, to compile and deploy your custom components. Do not modify this file; use `build.local.properties` to override any properties, if needed.

- `build.local.properties` – Allows you to specify which project you want to build and define properties specific to your environment. The main use of this file is declaring the project you want to build.
- `build.xml` – The Ant build script used to compile, build, and deploy your component. This file should not need modification.

Setting Up Your Project

To start developing your own plug-in:

1. Before you start, ensure you have the Java SDK and Apache Ant installed.
2. To create a new plug-in, create a new project directory in the `<PF_install_dir>/pingfederate/sdk/plugin-src` directory.
3. In the new project directory, create a subdirectory named `java`.
This is where you place the Java source code for your implementation(s).
Follow standard Java package and directory structure layout.
4. If your project depends on third-party libraries, create another subdirectory called `lib` and place the necessary JAR files in it.
5. The build script builds only one project at a time. Edit the `build.local.properties` file and set `target-plugin-name` to specify the name of the directory (under `<PF_install>/pingfederate/sdk/plugin-src`) that contains your project.
6. In `<PF_install>/pingfederate/sdk` run `ant` to display a list of available build targets:


```
[java] Main targets:
[java]
[java] clean-plugin      Clean the plug-in build directory
[java] deploy-plugin     Deploy the plug-in jar and libs to PingFederate
[java] jar-plugin       Package the plug-in jar
[java]
[java] Default target: help
```

Run the appropriate target to clean, build, or deploy your plug-in.

Note: Building the project with the `build.xml` included in the SDK is recommended since it packages the jars with additional metadata to make it discoverable by PingFederate. For detailed information, see [Identity Store Provisioner Implementation](#) on page 18.

Implementation Guidelines

The following sections provide specific programming guidance for developing custom interfaces. Note that the information is not exhaustive—consult the Javadocs to find more details about interfaces discussed here as well as additional functionality.

Shared Interfaces

All plug-in implementations generally invoke methods discussed in the following sections.

Configurable Plug-in

Any custom plug-in that requires UI settings is considered *configurable* and hence implements the `ConfigurablePlugin` interface. This ensures that PingFederate loads the plug-in instance with the correct configuration settings.

All plug-in types implement the `ConfigurablePlugin` interface and must define the following to enable configuration loading:

```
void configure(Configuration configuration)
```

During processing of a configurable plug-in instance, PingFederate calls the `ConfigurablePlugin.configure()` method and passes in a `Configuration` object. The `Configuration` object provides the plug-in adapter-instance configuration set by an administrator in the PingFederate UI.

The `sp-adapter-example` provided with the SDK shows how to use this method to initialize an adapter-instance from a saved configuration. Once your implementation loads the configuration values, the plug-in instance can use them in other method calls.

Describable Plug-in

Any plug-in that requires configuration screens in the PingFederate administrative console is considered a *describable* plug-in. Most plug-ins implement the `DescribablePlugin` interface to ensure that PingFederate renders the correct UI components based on the returned `PluginDescriptor`.

Adapter and custom data source plug-ins are a special case and do not implement the `DescribablePlugin` interface. However, they still return a plug-in descriptor (`AuthnAdapterDescriptor` and `SourceDescriptor` respectively) and are still considered describable plug-ins.

All describable plug-ins must define a UI descriptor. Use one of the following methods to implement a UI descriptor, depending on the type of plug-in:

- For `DescribablePlugin`:

```
PluginDescriptor getPluginDescriptor()
```

- For adapter plug-ins:

```
AuthnAdapterDescriptor getAdapterDescriptor()
```

- For custom data source plug-ins:

```
SourceDescriptor getSourceDescriptor()
```

In many cases, describable plug-ins return a subclass of `PluginDescriptor`, so the return type of the plug-in descriptor getters might be slightly different among plug-in implementations. Your plug-in implementation populates `PluginDescriptor` with `FieldDescriptors`, `FieldValidators`, and `Actions` and is presented as a set of UI components in the PingFederate administrative console.

Tip: Some plug-in types offer concrete descriptor implementations for developers. The Javadocs and examples provided with the SDK show which descriptor classes are available for each plug-in type. The examples also show you how to use `FieldDescriptors`, `FieldValidators`, and `Actions` directly to define your plug-in descriptor.

IdP Adapter Implementation

You create an IdP adapter by implementing the `IdpAuthenticationAdapter` or the `IdpAuthenticationAdapterV2` interface. The following Java packages are needed, at a minimum, for implementing this interface:

- `org.sourceid.saml20.adapter.idp.authn`
- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`

For each IdP adapter implementation, in addition to the methods described under [Shared Interfaces](#), you must define the following:

- Session Lookup
- Session Logout

IdP Adapter Session Lookup

```
java.util.Map lookupAuthN(javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse resp,
    java.lang.String partnerSpEntityId,
    AuthnPolicy authnPolicy,
    java.lang.String resumePath)
    throws AuthnAdapterException, java.io.IOException
```

`PingFederate` invokes the `lookupAuthN()` method of your IdP adapter to look up user-session information to handle a request. This method is invoked regardless of whether the request is for IdP- or SP-initiated SSO, an OAuth transaction, or direct IdP-to-SP adapter processing.

Note: The `IdpAuthenticationAdapterV2` interface provides an overloaded version of `lookupAuthN()` applicable to `PingFederate` versions 6.4 and higher. Use this interface if your adapter requires additional parameters from `PingFederate`. Refer to the `IdpAuthenticationAdapterV2` interface in the Javadocs for a complete list of available parameters.

In most implementations, a user's session information or a reference to it is communicated to `PingFederate` via the `HttpServletRequest`, which is passed to the `lookupAuthN()` method. For example, the user's session information can be passed in by the IdP application as a cookie or query parameter.

If the request from the user's browser does not contain the necessary information to identify the user, you can use the `HttpServletResponse` in various ways to retrieve the user's session data—for example, by creating a 302 redirect or presenting a Web page asking for credentials. If your adapter implementation uses the `HttpServletResponse` to retrieve the user's session information, you must return the user's browser to the URL in the `resumePath` parameter set by the `PingFederate` runtime server and passed to

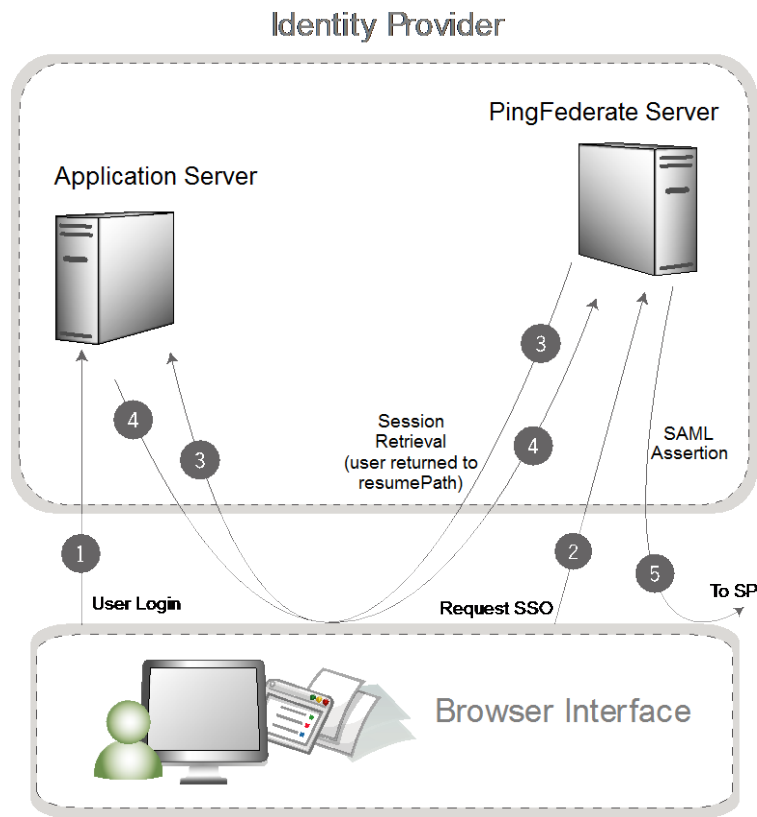
this method. The `resumePath` is a relative URL signaling PingFederate that a user is continuing an SSO transaction that has already been initiated.

Tip: When creating a custom adapter, you can design it to render a template for processing and returning HTML to the user's browser using the `TemplateRendererUtil`. A sample (`template-render-adapter-example`) is included in the `sdk/plugin-src` directory of your PingFederate instance.

If your adapter implementation writes to the `HttpServletResponse` to retrieve the user's session data, we recommend that the browser return to the `resumePath` URL at all times, whether the retrieval succeeds or fails. Doing so ensures the adapter does not interrupt the "adapter chain" if it is used with the Composite Adapter. The Composite Adapter allows an administrator to "chain" together a selection of available adapter instances for a connection. At runtime, adapter chaining means that SSO requests are passed sequentially through each adapter instance until one or more authentication results are found for the user. If the browser is unable to return to the `resumePath` URL at all times, then it could interrupt the adapter chain causing unexpected results for the Composite Adapter.

For some authentication mechanisms, not all adapters can return the browser to the `resumePath` URL. Such adapters should not be used with the Composite Adapter's "Sufficient" chaining policy (see Composite Adapter Configuration in the PingFederate *Administrator's Manual*).

The following diagram illustrates the request sequence of an IdP-initiated SSO scenario that uses the `resumePath`:



Processing Steps

1. User logs in to a local application or domain through an identity-management system or some other authentication mechanism.
2. User clicks a link or otherwise requests access to a protected resource located in the SP domain. The link or other mechanism invokes the PingFederate SSO service.
3. PingFederate invokes the designated adapter's lookup method, including the `resumePath` parameter. In this example, the adapter determines there is not enough information and redirects the browser to the application server to fetch additional session information.
4. The application server returns the session information and redirects the browser along with the returned information to `resumePath` URL.
5. PingFederate generates a SAML assertion and sends the browser with the SAML assertion to the SP's SAML gateway.

IdP Adapter Session Logout

```
boolean logoutAuthN(java.util.Map authnIdentifiers,
    javax.servlet.http.HttpServletRequest req,
    javax.servlet.http.HttpServletResponse resp,
    java.lang.String resumePath)
    throws AuthnAdapterException, java.io.IOException
```

During SLO request processing, PingFederate invokes your IdP adapter's `logoutAuthN()` method to terminate a user's session. This method is invoked during IdP- or SP-initiated SLO requests.

Like the `lookupAuthN()` method, the `logoutAuthN()` method has access to the user's `HttpServletRequest` and `HttpServletResponse` objects. Use these objects to retrieve data about the user's session as well as to redirect the browser to an endpoint used to terminate the session at the application. Again, the `resumePath` parameter contains the URL to which the user is redirected to complete the SLO process.

SP Adapter Implementation

You create an SP adapter by implementing the `SPAuthenticationAdapter` interface. The Java packages required are, at a minimum:

- `org.sourceid.saml20.adapter.sp.authn`
- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`

At a high level, in addition to the methods described under [Shared Interfaces](#), you must define the following:

- Session Creation
- Session Logout
- Account Linking (if configured in PingFederate for an IdP partner)

SP Session Creation

```
java.io.Serializable createAuthN(SsoContext ssoContext,  
    javax.servlet.http.HttpServletRequest req,  
    javax.servlet.http.HttpServletResponse resp,  
    java.lang.String resumePath)
```

PingFederate invokes the `createAuthN()` method during the processing of an SSO request to establish a security context in the external application for the user. This method is similar to the `IdpAuthenticationAdapter.lookupAuthN()` method in terms of the objects passed to it and its support for asynchronous requests via the `HttpServletResponse` and `resumePath` parameters. This method also accepts an `SsoContext` object, which has access to information such as user attributes and the target destination URL.

SP Adapter Session Logout

```
boolean logoutAuthN(java.io.Serializable authnBean,  
    javax.servlet.http.HttpServletRequest req,  
    javax.servlet.http.HttpServletResponse resp,  
    java.lang.String resumePath)  
    throws AuthnAdapterException, java.io.IOException
```

PingFederate invokes the `logoutAuthN()` method during an SLO request to terminate a user's session with the external application. The `HttpServletResponse` and `resumePath` objects are available to support scenarios where redirection of the user's browser is needed to an additional service to clean up any remaining sessions.

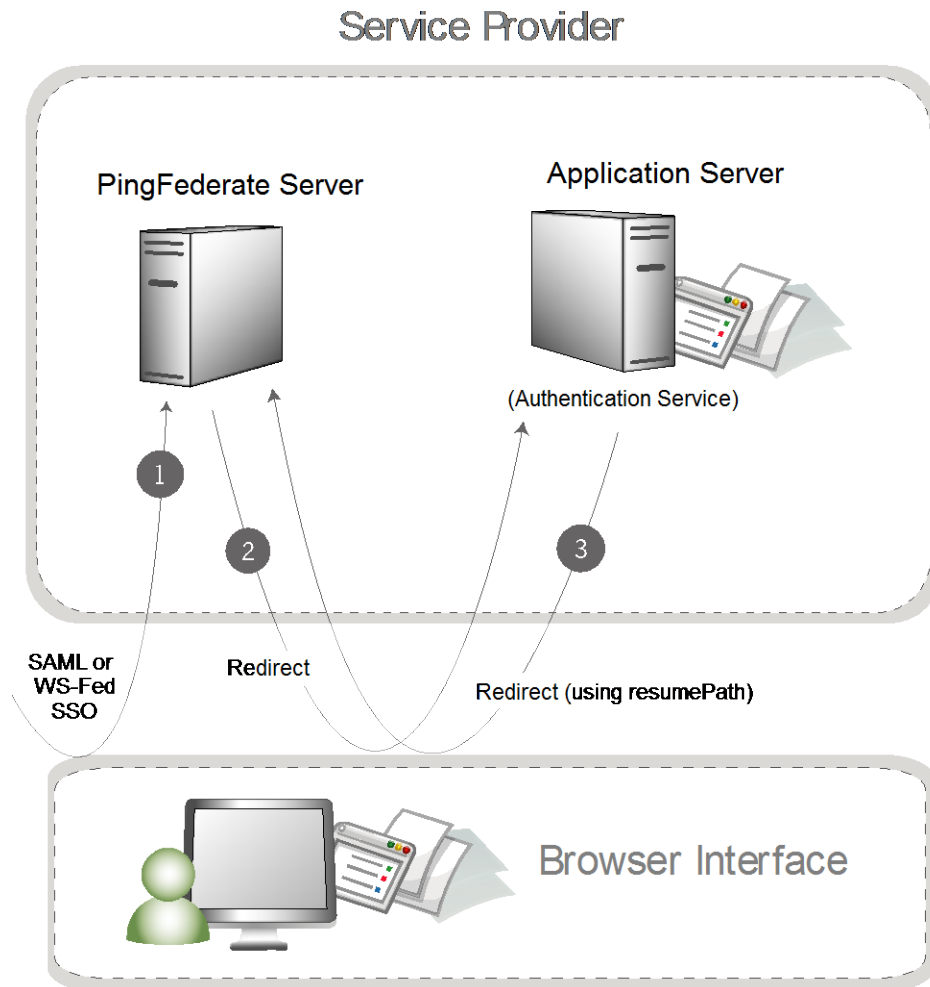
SP Account Linking

```
java.lang.String lookupLocalUserId(  
    javax.servlet.http.HttpServletRequest req,  
    javax.servlet.http.HttpServletResponse resp,  
    java.lang.String partnerIdpEntityId,  
    java.lang.String resumePath)  
    throws AuthnAdapterException, java.io.IOException
```

PingFederate invokes the `lookupLocalUserId()` method during an SSO request when the IdP connection is configured to use account linking but no account link for this user is yet established. Once the account link is set, PingFederate maintains this information until the user "defederates." Defederation occurs when the user clicks a link redirecting him/her to the `/sp/defederate.ping` PingFederate endpoint.

The `HttpServletResponse` and `resumePath` objects are used to send the user to a local service where the user authenticates. After authentication, the user is redirected to the URL specified in the `resumePath` parameter and PingFederate completes the account link.

The following diagram illustrates a typical account-link sequence:



Use the `HttpServletRequest` to read a local session token. The `String` object returned from the `lookupLocalUserId()` method should be a local user identifier.

Token Processor Implementation

You create a token-processor implementation (for PingFederate 6.0 and higher) by implementing the `TokenProcessor` interface. The following Java packages are needed, at a minimum, for implementing this interface:

- `org.sourceid.saml20.adapter.attribute`
- `org.sourceid.saml20.adapter.idp.authn`
- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`
- `org.sourceid.wstrust.model`
- `org.sourceid.wstrust.plugin`
- `org.sourceid.wstrust.plugin.process`

- `com.pingidentity.sdk`

For each token-processor implementation, in addition to the methods described under [Shared Interfaces](#), you must define the method:

```
TokenContext processToken(T token)
```

PingFederate invokes the `processToken()` method during the processing of an STS request to perform necessary operations for determining the validity of a token. Type `T` must extend, at a minimum, the type `SecurityToken`. The type `BinarySecurityToken` is also available and may be used to represent custom security tokens that can be transported as Base64-encoded data.

Token Generator Implementation

You create a token-generator implementation (for PingFederate 6.0 and higher) by implementing the `TokenGenerator` interface. The following Java packages needed, at a minimum, for implementing this interface:

- `org.sourceid.saml20.adapter.sp.authn`
- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`
- `org.sourceid.wstrust.model`
- `org.sourceid.wstrust.plugin`
- `org.sourceid.wstrust.plugin.process`
- `com.pingidentity.sdk`

For each token-generator implementation, described under [Shared Interfaces](#), you must define the method:

```
SecurityToken generateToken(TokenContext attributeContext)
```

PingFederate invokes the `generateToken()` method during the processing of an STS request to perform necessary operations for generation of a security token. The type `BinarySecurityToken` is available and may be used to represent custom security tokens that can be transported as Base64-encoded data. The `TokenContext` contains subject data available for insertion into the generated security token.

Authentication Selector Implementation

Authentication selectors allow PingFederate (version 6.6 and higher) to choose an appropriate authentication source, an IdP adapter or an IdP connection (for federation hub use cases), based on criteria defined in the authentication selector instance.

When creating an authentication selector, the following are the primary Java packages used:

- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`
- `com.pingidentity.sdk`

For each authentication selector implementation, in addition to the methods described under [Shared Interfaces](#), you must define the following at a minimum:

- Context Selection
- Authentication Selector Callback

Context Selection

```
AuthenticationSelectorContext selectContext(HttpServletRequest req,
    HttpServletResponse resp,
    Map<AuthenticationSourceKey, String> mappedAuthnSourcesNames,
    Map<String, Object> extraParameters,
    String resumePath)
```

PingFederate calls the `selectContext()` method to determine which authentication source to select. The `mappedAuthnSourcesNames` contains the list of `AuthenticationSourceKeys` and names that are available for the selector to reference. The `HttpServletRequest` is available to evaluate cookies, parameters, headers, etc. to help determine which authentication source should be selected. The `HttpServletResponse` is also available if the authentication selector requires user interaction to help determine the appropriate authentication source to select. If the `resp` object is written to, it is considered a committed response and returned to the user's browser. The `resumePath` is a relative URL that should be used in conjunction with the `resp` object, such that the user's browser can be sent to this URL to resume the SSO workflow.

Once an authentication source is selected, an `AuthenticationSelectorContext` can be created to denote which authentication source to use. The selected authentication source can be referenced by its ID or by its context. The context is a name that decouples authentication selectors from the configured IDs.

Authentication Selector Callback

```
void callback(HttpServletRequest req,
    HttpServletResponse resp,
    Map authnIdentifiers,
    AuthenticationSourceKey authenticationSourceKey,
    AuthenticationSelectorContext authnSelectorContext);
```

PingFederate calls the `callback()` method after a selected authentication source is authenticated against. The `callback()` method allows authentication selectors to update resulting attributes, set cookies, or perform other custom functions.

Note: Writing content to the `resp` object in the `callback()` method is not supported, and doing so may result in unexpected behavior. Setting cookies is acceptable.

Custom Data Source Implementation

Out of the box, PingFederate provides the capability of querying data sources for a variety of purposes using LDAP or JDBC interfaces. You can use the PingFederate SDK to build data source connectors to query additional data source types. Examples of other data sources include a Web service, a flat file, or perhaps a different way of using a JDBC or LDAP connection than what is supplied by PingFederate.

The following are the primary Java packages used to build a custom data source:

- `com.pingidentity.sources`
- `com.pingidentity.sources.gui`

For each implementation, described under [Shared Interfaces](#), you must define the following at a minimum:

- Connection Testing
- Available Fields Retrieval
- Data Source Query Handling

Data Source Connection Testing

```
boolean testConnection()
```

When associating a custom data source with an IdP or SP connection, PingFederate tests connectivity to the data source by calling the `testConnection()` method. Your implementation of this method should perform the necessary steps to demonstrate a successful connection and return `true`. Return `false` if your implementation cannot communicate with the data store. A `false` result prevents an administrator from continuing with the data source configuration.

Data Source Available Fields Retrieval

```
java.util.List<java.lang.String> getAvailableFields()
```

PingFederate calls the `getAvailableFields()` method to determine the available fields that could be returned from a query of this data source. These fields are displayed to the PingFederate administrator during the configuration of a data source lookup. The administrator can then select the attributes from the data source and map them to the adapter or attribute contract. PingFederate requires at least one field returned from this method.

Data Source Query Handling

```
java.util.Map<java.lang.String, java.lang.Object> retrieveValues(  
    java.util.Collection<java.lang.String> attributeNamesToFill,  
    SimpleFieldList filterConfiguration)
```

When processing a connection using a custom data source, PingFederate calls the `retrieveValues()` method to perform the actual query for user attributes. This method receives a list of attribute names that should be populated with data. The method may also receive a `filterConfiguration` object populated with a list of fields. Each field contains a name/value pair that is determined at runtime and collectively used as the criteria for selecting a specific record. In most cases, the criteria are used to locate additional user attributes.

You create the filter criteria selections needed for this lookup by passing back a `CustomDataSourceDriverDescriptor`, an implementation of `SourceDescriptor`, from the `getSourceDescriptor()` method. A `CustomDataSourceDriverDescriptor` can include a `FilterFieldDataDescriptor` composed of a list of fields that can be used as the query criteria. This list of fields is displayed similarly to the other UI-descriptor display fields.

Note: The `filterConfiguration` object is set and populated with a list of fields only if the data source was defined with a `CustomDataSourceDriverDescriptor`. If the `CustomDataSourceDriverDescriptor` was not used in the definition of the data source, the `filterConfiguration` object is set to null.

Important: To pass runtime attribute values to the filter, an administrator must reference the attributes using the `#{attribute name}` format when defining a filter in the PingFederate administrative console.

Once all the relevant attributes are retrieved from the data source, they must be returned as a map of name/value pairs, where the names correspond to the initial collection of attribute names that was passed into the method and the values are the attributes.

Password Credential Validator Implementation

Password credential validators allow PingFederate administrators to define a centralized location for username/password validation, allowing validator instances to be referenced by various PingFederate configurations..

To implement a custom password credential validator, the following Java packages need to be imported:

- `org.sourceid.saml20.adapter.gui`
- `org.sourceid.saml20.adapter.conf`
- `org.sourceid.util.log`
- `com.pingidentity.sdk`
- `com.pingidentity.sdk.password`

For each implementation, in addition to the methods described under [Shared Interfaces](#), you must define the following at a minimum:

```
AttributeMap processPasswordCredential(String username,  
    String password)  
    throws PasswordValidationException
```

This method takes a `username` and `password` and verifies the credential against an external source. If the credentials are valid, then an `AttributeMap` is returned containing at least one entry representing the principal. If the credentials are invalid, then `null` or an empty map is returned. A `PasswordValidationException` is thrown if the plug-in was unable to validate the credentials (for example, due to an offline host or network problems).

To enable password changes in a password credential validator, implement the `com.pingidentity.sdk.password.ChangeablePasswordCredential` interface.

Note: Depending on your password management system, additional system configuration may be necessary to enable password changes—for example, passwords can be changed in Active Directory only if SSL is enabled.

Identity Store Provisioner Implementation

You create an Identity Store Provisioner by implementing either the `IdentityStoreUserProvisioner` or `IdentityStoreProvisioner` interface.

The former supports provisioning and deprovisioning users to an external user store; the latter adds the capability to support provisioning and deprovisioning of groups.

Implementing the `IdentityStoreUserProvisioner` Interface

Implement the `IdentityStoreUserProvisioner` interface to provision and deprovision users to an external user store.

Tip: The `IdentityStoreUserProvisioner` interface does not provision or deprovision groups. For group support, see [Implementing the `IdentityStoreProvisioner` Interface](#).

The following Java packages are needed, at a minimum, for implementing this interface:

- `com.pingidentity.sdk.provision`
- `com.pingidentity.sdk.provision.exception`
- `com.pingidentity.sdk.provision.users.request`
- `com.pingidentity.sdk.provision.users.response`

For each Identity Store Provisioner implementation, in addition to the methods described under [Shared Interfaces](#), you must implement the following:

- Create User
- Read User
- Update User
- Delete User

Create User

```
UserResponseContext createUser(CreateUserRequestContext createRequestCtx)
    throws IdentityStoreException
```

`PingFederate` invokes the `createUser()` method of your Identity Store Provisioner in response to create-user requests made to `PingFederate` services, for example Inbound Provisioning. This method is responsible for creating the user in the user store managed by the Identity Store Provisioner.

The `CreateUserRequestContext` will contain all information needed to fulfill the request, e.g. user attributes. If the user was successfully provisioned, a `UserResponseContext` should be returned and contain the user attributes used to provision the user. An `IdentityStoreException` should be thrown if an error occurred during the creation process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Read User

```
UserResponseContext readUser(ReadUserRequestContext readRequestCtx)
    throws IdentityStoreException
```


PingFederate invokes the `readUser()` method of your Identity Store Provisioner in response to get-user requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for retrieving user data from the user store managed by the Identity Store Provisioner.

The `ReadUserRequestContext` will contain all information needed to fulfill the request, e.g. user id. If the user data was successfully retrieved, a `UserResponseContext` should be returned and contain the user attributes for the user. An `IdentityStoreException` should be thrown if an error occurred during the retrieval process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Update User

```
UserResponseContext updateUser(UpdateUserRequestContext updateRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `updateUser()` method of your Identity Store Provisioner in response to update-user requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for updating the user in the user store managed by the Identity Store Provisioner.

The `UpdateUserRequestContext` will contain all information needed to fulfill the request, e.g. user attributes. If the user data was successfully updated, a `UserResponseContext` should be returned containing the user's updated attributes. An `IdentityStoreException` should be thrown if an error occurred during the update process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Delete User

```
void deleteUser(DeleteUserRequestContext deleteRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `deleteUser()` method of your Identity Store Provisioner in response to delete-user requests made to PingFederate services, such as Inbound Provisioning. This method is responsible for deprovisioning the user in the user store managed by the Identity Store Provisioner.

The `DeleteUserRequestContext` will contain all information needed to fulfill the request, e.g. user id. An `IdentityStoreException` should be thrown if an error occurred during the deprovision process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Note: The plugin implementation for delete MAY choose not to permanently delete the resource, but MUST return a `NotFoundException` for all `readUser()`, `updateUser()`, and `deleteUser()` operations associated with the previously deleted Id. In addition, the plugin MUST not consider the deleted user in conflict calculation. For example, a `createUser()` request for a user with a previously deleted ID should NOT throw a `ConflictException`.

Implementing the IdentityStoreProvisioner Interface

Implement the `IdentityStoreProvisioner` interface to provision and deprovision users and (optionally) groups to an external user store. The following Java packages are needed, at a minimum, for implementing this interface:

- `com.pingidentity.sdk.provision`
- `com.pingidentity.sdk.provision.exception`

- `com.pingidentity.sdk.provision.users.request`
- `com.pingidentity.sdk.provision.users.response`
- `com.pingidentity.sdk.provision.groups.response`
- `com.pingidentity.sdk.provision.groups.request`

Note: Group support is optional (see [Check for Group Provisioning Support](#)).

For each Identity Store Provisioner implementation, in addition to the methods described under [Shared Interfaces](#), you must implement the following:

- Create User
- Read User
- Update User
- Delete User
- Check for Group Provisioning Support
- Create Group
- Read Group
- Update Group
- Delete Group

Create User

```
UserResponseContext createUser(CreateUserRequestContext createRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `createUser()` method of your Identity Store Provisioner in response to create-user requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for creating the user in the user store managed by the Identity Store Provisioner.

The `CreateUserRequestContext` will contain all information needed to fulfill the request, e.g. user attributes. If the user was successfully provisioned, a `UserResponseContext` should be returned and contain the user attributes used to provision the user. An `IdentityStoreException` should be thrown if an error occurred during the creation process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Read User

```
UserResponseContext readUser(ReadUserRequestContext readRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `readUser()` method of your Identity Store Provisioner in response to get-user requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for retrieving user data from the user store managed by the Identity Store Provisioner.

The `ReadUserRequestContext` will contain all information needed to fulfill the request, e.g. user id. If the user data was successfully retrieved, a `UserResponseContext` should be returned and contain the user attributes for the user. An `IdentityStoreException` should be thrown if an error occurred during

the retrieval process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Update User

```
UserResponseContext updateUser(UpdateUserRequestContext updateRequestCtx)
throws IdentityStoreException
```

PingFederate invokes the `updateUser()` method of your Identity Store Provisioner in response to update-user requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for updating the user in the user store managed by the Identity Store Provisioner.

The `UpdateUserRequestContext` will contain all information needed to fulfill the request, e.g. user attributes. If the user data was successfully updated, a `UserResponseContext` should be returned containing the user's updated attributes. An `IdentityStoreException` should be thrown if an error occurred during the update process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Delete User

```
void deleteUser(DeleteUserRequestContext deleteRequestCtx)
throws IdentityStoreException
```

PingFederate invokes the `deleteUser()` method of your Identity Store Provisioner in response to delete-user requests made to PingFederate services, such as Inbound Provisioning. This method is responsible for deprovisioning the user in the user store managed by the Identity Store Provisioner.

The `DeleteUserRequestContext` will contain all information needed to fulfill the request, e.g. user id. An `IdentityStoreException` should be thrown if an error occurred during the deprovision process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Note: The plugin implementation for delete MAY choose not to permanently delete the resource, but MUST return a `NotFoundException` for all `readUser()`, `updateUser()`, and `deleteUser()` operations associated with the previously deleted id. In addition, the plugin MUST not consider the deleted user in conflict calculation. For example, a `createUser()` request for a user with a previously deleted ID should NOT throw a `ConflictException`.

Check for Group Provisioning Support

```
boolean isGroupProvisioningSupported()
throws IdentityStoreException
```

Implement this `isGroupProvisioningSupported()` method to return true if group provisioning is supported by your Identity Store Provisioner or false otherwise. An `IdentityStoreException` should be thrown if an error occurred during the query process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Create Group

```
GroupResponseContext createGroup(CreateGroupRequestContext createRequestCtx)
throws IdentityStoreException
```

PingFederate invokes the `createGroup()` method of your Identity Store Provisioner in response to create-group requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for creating the group in the user store managed by the Identity Store Provisioner if the `isGroupProvisioningSupported()` returns true; otherwise, it should throw `NotImplementedException`.

The `CreateGroupRequestContext` will contain all information needed to fulfill the request, e.g. the group attributes. If the group was successfully provisioned, a `GroupResponseContext` should be returned and contain the group attributes used to provision the group. An `IdentityStoreException` should be thrown if an error occurred during the creation process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Read Group

```
GroupResponseContext readGroup(ReadGroupRequestContext readRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `readGroup()` method of your Identity Store Provisioner in response to get-group requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for retrieving user data from the user store managed by the Identity Store Provisioner if the `isGroupProvisioningSupported()` returns true; otherwise, it should throw `NotImplementedException`.

The `ReadGroupRequestContext` will contain all information needed to fulfill the request, e.g. group id. If the user data was successfully retrieved, a `GroupResponseContext` should be returned and contain the group attributes for the group. An `IdentityStoreException` should be thrown if an error occurred during the retrieval process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Update Group

```
GroupResponseContext updateGroup(UpdateGroupRequestContext updateRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `updateGroup()` method of your Identity Store Provisioner in response to update-group requests made to PingFederate services, for example Inbound Provisioning. This method is responsible for updating the group in the user store managed by the Identity Store Provisioner if the `isGroupProvisioningSupported()` returns true; otherwise, it should throw `NotImplementedException`.

The `UpdateGroupRequestContext` will contain all information needed to fulfill the request, e.g. group attributes. If the group data was successfully updated, a `GroupResponseContext` should be returned containing the group's updated attributes. An `IdentityStoreException` should be thrown if an error occurred during the update process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Delete Group

```
void deleteGroup(DeleteGroupRequestContext deleteRequestCtx)
    throws IdentityStoreException
```

PingFederate invokes the `deleteGroup()` method of your Identity Store Provisioner in response to delete-group requests made to PingFederate services, such as Inbound Provisioning. This method is

responsible for deprovisioning the group in the user store managed by the Identity Store Provisioner if the `isGroupProvisioningSupported()` returns true; otherwise, it should throw `NotImplementedException`.

The `DeleteGroupRequestContext` will contain all information needed to fulfill the request, e.g. group id. An `IdentityStoreException` should be thrown if an error occurred during the deprovision process. See `com.pingidentity.sdk.provision.exception` package for different exceptions that can be thrown.

Building and Deploying Your Project

To build and deploy your project, you can choose to use the provided Apache Ant script or another build utility.

Building and Deploying With Ant

The PingFederate Java SDK comes with an Apache Ant build script that makes building and deploying your project simple.

1. Edit the `build.local.properties` file and set the `target-plugin.name` property to the name of your project subdirectory (see [Directory Structure](#) on page 6).

Note: You can develop source code for multiple projects simultaneously, but you can build and deploy only one at a time. Change the value of the `target-plugin.name` property as needed to build and deploy other projects.

2. If your project depends on any third-party jars, place them into your project's `lib` directory.

If the directory does not exist, create a new directory called `lib`, directly under your project's directory, For example, `pingfederate/sdk/plugin-src/<subproject-name>/lib`

3. On the command line in the `sdk` directory, use `ant` to clean, build, and package or to build, package, and deploy your project.

To clean the project, enter:

```
ant clean-plugin
```

To compile the project, enter:

```
ant compile-plugin
```

To compile the project and create a JAR, enter:

```
ant jar-plugin
```

The SDK creates deployment descriptor(s) in the `PF_INF` directory and places it in a JAR. The descriptor tells PingFederate what plug-in implementations are contained in the JAR.

The compiled class files and the deployment descriptor(s) are placed in the `pingfederate/sdk/plugin-src/<subproject-name>/build/classes` directory.

The `pf.plugins.<subproject-name>.jar` file is placed in the `pingfederate/sdk/plugin-src/<subproject-name>/build/jar` directory.

To compile, create a JAR, and deploy the project to PingFederate, enter:

```
ant deploy-plugin
```

This build target performs the steps described above as well as deploying any JAR files found in the `lib` directory of your subproject.

Note: To deploy your plug-in manually to an installation of the PingFederate server, copy the JAR file and any third-party JAR files into the `/server/default/deploy/` directory of that PingFederate installation.

4. Restart the PingFederate server.

Manually Building and Deploying

To build your project with another build utility, you must take some prerequisite steps to create the deployment descriptors for each of your plug-ins. The deployment descriptor files allow PingFederate to discover your plug-ins.

Creating Deployment Descriptors:

1. In your project, create a new directory called `PF-INF`. This directory must be at the root of your JAR file, similar to `META-INF`.
2. Inside `PF-INF` create the appropriate text file(s) for each type of plug-ins you created:

Plug-in Type	Filename
IdP Adapter	idp-authn-adapters
SP Adapter	sp-authn-adapters
Custom Data Source	custom-drivers
Token Processor	token-processors
Token Generator	token-generators
Authentication Selector	authentication-selectors
Password Credential Validator	password-credential-validators
Identity Store Provisioner	identity-store-provisioners

3. In each text file created, specify the fully qualified class name of each plug-in that implements the corresponding plug-in interface. Place each class name on a separate line.

Manually Building Your Project:

To compile your project, you need to have the following directories on your classpath:

- `pingfederate/server/default/lib`
- `pingfederate/lib`
- `pingfederate/sdk/lib`
- `pingfederate/sdk/plugin-src/<subproject-name>/lib`

To create a JAR, simply archive the compiled class files along with the deployment descriptor(s) using your build tool. The deployment descriptors must be in the `PF-INF` directory, located at the root of the JAR.

Deploying Your Project:

To deploy your plug-in, simply copy the JAR file and any third-party JAR files into the `pingfederate/server/default/deploy` directory of the PingFederate installation.

Logging

You can use a typical logging pattern based on the Apache Commons logging framework to log messages from your adapter, token translator, or custom data source driver. The SP adapter contained in the directory `sdk/adapters-src/sp-adapter-example` shows how to use a logger for your adapter.

PingFederate[®]

Upgrade Utility

Version 7.3

User Guide



© 2015 Ping Identity® Corporation. All rights reserved.

PingFederate Upgrade Utility *User Guide*
Version 7.3
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909
Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: **January 29, 2015**.

Contents

Upgrade Introduction	4
Upgrading PingFederate	4
Step One -- Preparation	4
Step Two -- Run the Upgrade Utility	5
Step Three -- Copy Any Customized Files or Settings	8
Known Limitations	9

Upgrade Introduction

This Upgrade Utility is a command-line tool that automatically migrates existing PingFederate 6.x (and higher) configurations to the current version. The tool first installs the new distribution and then copies over files or property values from the previous installation.

As an option, you can run the utility with a *custom* switch that allows you to choose whether to override any new system defaults or to install newer versions of bundled adapters (see [Using Custom Mode](#) on page 7).

Note: The Upgrade Utility does *not* affect the existing installation of PingFederate. After you complete the steps in this Guide and then test and deploy the new PingFederate in production, we recommend that you remove the previous installation.

The Upgrade Utility creates a detailed log showing what files and settings are affected during the upgrade.

If you are upgrading from PingFederate 5.x, contact support@pingidentity.com for more information.

Intended Audience

The PingFederate Upgrade Utility and this document are intended for system administrators with significant knowledge of the current PingFederate configuration and enterprise deployment.

Upgrading PingFederate

Fully upgrading your existing PingFederate installation involves initial preparation before running the Upgrade Utility. Then afterward, some manual copying may be needed for customized files or supporting Web-application deployments.

The Upgrade Utility is located in the `/bin` directory of the distribution ZIP as either a batch file (for Windows installations) or a shell script (for Linux/Unix installations).

Note: The Upgrade Utility does not migrate OAuth client configurations created prior to PingFederate version 6.8. Reconfigure clients manually using either the PingFederate Client Management screen or the REST-based Web Service for OAuth client management (see Client Management in the PingFederate *Administrator's Manual*).

Important: As of PingFederate version 6.11, security updates related to XML encryption may affect existing partner connections, requiring manual reconfiguration or partner notification. For more information, see Upgrade Considerations in the PingFederate Release Notes.

Step One -- Preparation

Before running the Upgrade Utility:

1. If you have not already done so, download the current version of PingFederate and obtain a new license key.

2. If you have not recently updated the Java SE Development Kit (JDK) to version 7 or 8 on your PingFederate host machine(s), you must do so.

Note: When you upgrade the JDK, be sure to modify the previously defined paths for the system `JAVA_HOME` and `PATH` environment variables. (For more information, see *Installing the JDK* in *Getting Started*.)

Important: PingFederate versions prior to 6.7 will not start using Java 7 or 8. If you need to start the previous PingFederate version on the same server after the upgrade, retain the older Java installation and change environment variables back when needed.

3. (Optional) Complete any unfinished connections (“Drafts”) in the existing PingFederate administrative console, *if* you want to include them in the migration.

Because they may cause unpredictable errors in the administrative console, connection drafts cannot be moved from one instance of PingFederate to another.

4. Unzip the Upgrade Utility distribution into a directory on the same machine running PingFederate.
5. If you are upgrading PingFederate in a clustered-server environment, unzip the Upgrade Utility on each PingFederate runtime host.

Note: Start the upgrade on the PingFederate server running the administrative console.

Step Two -- Run the Upgrade Utility

1. Stop the currently installed PingFederate administrative server.
2. From the `pf-upgrade-7.3.0/bin` directory at a system prompt, execute the following command:

On Windows:

```
upgrade <sourcePingfederateRootDir> <outputDir> <pingfederateZip> [-c]
```

On Linux/Unix:

```
./upgrade.sh <sourcePingfederateRootDir> <outputDir> <pingfederateZip> [-c]
```

where:

- `<sourcePingfederateRootDir>` is the full or relative path of the base directory where the existing PingFederate software (`pingfederate/*`) is installed;

Important: The `pingfederate` subdirectory *must* exist by that name for the upgrade utility to function correctly.

- `<outputDir>` is the full or relative path of the directory that will contain the new PingFederate base directory;
- `<pingfederateZip>` is the full or relative path and file name for the current distribution; and
- `-c` runs the utility in optional custom mode (see [Using Custom Mode](#) on page 7).

Examples:

```
upgrade c:\sso\pingfederate-6.0.0 c:\sso c:\sso\pingfederate-7.3.0.zip  
./upgrade.sh /sso/pingfederate-6.0.0 /sso /sso/pingfederate-7.3.0.zip
```

For version 7.3, this command would install the new software into:

```
c:\sso\pingfederate-7.3.0 (on Windows)  
/sso/pingfederate-7.3.0 (on Linux/Unix)
```

Note: This directory is referred to in this document as `<pf_install>`, with subdirectory separators standardized as forward slashes (/).

After a moment, the command window displays messages indicating upgrade progress. The process is complete when this message appears:

```
Upgrade completed with [N] errors and [N] warnings
```

If there are errors, scroll up the command window to see them and correct indicated problems. Errors during the upgrade should be rare but may include such input/output problems as missing or malformed configuration files in the source installation.

The messages, including any errors, are also logged to the Upgrade Utility base directory in the file:

```
upgrade.log
```

Tip: Rerun the Upgrade Utility as many times as needed to correct any problems.

3. Copy the new PingFederate license file, `pingfederate.lic`, into the directory:

```
<pf_install>/pingfederate/server/default/conf
```

Tip: Alternatively, you can import the new license via the administrative console at the next step (see *Installing a New License Key in the PingFederate Administrator's Manual*).

4. Start the new PingFederate installation and open the administrative console.
5. Verify configuration settings in the administrative console.

Note: Access the connection management screens and ensure that automatic validation is run on each connection.

6. If you are upgrading a server cluster:
 - a. Stop each server node and run the Upgrade Utility for each PingFederate installation in the cluster.
 - b. Start the *new* installation on each node.

- c. In the administrative console, click **Cluster Management** on the Main Menu.
- d. On the Cluster Management screen, ensure all nodes are shown in the list of IP addresses.
- e. Click **Replicate Configuration**.

Important: The Upgrade Utility *does not* migrate the administrative user-interface (UI) configuration to clustered servers. For more information about cluster management, see the Server Clustering Guide.

7. If PingFederate is running as a service, stop the existing service and re-install it (for Windows) or reconfigure it (for Linux/Unix) to use the new PingFederate installation.

For more information, see Installation in *Getting Started*.

8. If applicable, repeat the previous step for each server node in a clustered deployment (update each PingFederate service).
9. If the previous PingFederate installation (the source) has OAuth use cases, run the table-setup script, `access-grant-attribute-<database>.sql`, found in the `<pf_install>/pingfederate/server/default/conf/access-grant/sql-scripts` directory for your database server. For more information, see [Defining an OAuth Grant Data Store](#) in the *Administrator's Manual*.

Using Custom Mode

The custom-mode feature (invoked with the `-c` option on the command line) allows administrators to override several newer default security settings (new, depending on which PingFederate version is currently running). In addition, if the installed OpenToken adapter is out of date, running the tool in custom mode allows you to replace the adapter with the latest version, if applicable.

About Security Defaults

In general, using the new security defaults is highly recommended and should not cause significant issues for most PingFederate installations. The newer default security settings include:

- Disabling weaker cipher suites for both the SUN and LUNA Java Cryptography Extension (JCE) in PingFederate version 6.2 and later. If you want to see which cipher suites are commented out, choose yes (y) when prompted on whether to use the new defaults. Then, after the upgrade is complete, refer to either of the following configuration files in the new installation's `<pf_install>/pingfederate/server/default/data/config-store` directory:
 - `com.pingidentity.crypto.SunJCEManager.xml`
 - `com.pingidentity.crypto.LunaJCEManager.xml`
- Disabling SQLFilter as of PingFederate version 6.2.
- Disabling the sending of PingFederate session cookies by way of the unsecure hypertext transfer protocol. As of version 6.5, the session cookie is configured to be sent only using HTTPS.

Upgrading Adapters

Upgrading the OpenToken adapter from previous versions is also recommended and will not normally require any follow-on configuration changes. However, if your existing installation uses a version of the

OpenToken adapter prior to 2.3, upgrading requires minor configuration modifications in the PingFederate console as well as redeployment of the agent configuration file (see OpenToken Adapter Configuration in the PingFederate *Administrator's Manual*).

If you are upgrading from an OpenToken version prior to 2.5.1, we recommend that you redeploy agent configuration files, if applicable, as well as any new agent libraries contained in recent versions of PingFederate integration kits and other plug-ins that use OpenToken.

Note: Starting in PingFederate 7.2, the LDAP Java Adapter is no longer supported. This adapter was deprecated in PingFederate 6.6 and replaced by the LDAP Password Credential Validator (PCV), which can be used with the HTML Form or HTTP Basic IdP Adapters. For more information, see HTTP Basic Adapter Configuration and HTML Form Adapter Configuration in the PingFederate *Administrator's Manual*.

Step Three -- Copy Any Customized Files or Settings

If you have modified any user-facing Velocity templates, you must copy the HTML files over to the new installation manually for each server node (see Customizing User-Facing Screens in the PingFederate *Administrator's Manual*). The templates are located in the directory:

```
<pf_install>/pingfederate/server/default/conf/template
```

Caution: Supporting CSS and image file names were changed as of PingFederate 7.0. For each modified HTML template copied, add `.1` to the base name for each CSS file referenced in the header.

Example: `<link rel="..." href="assets/css/screen.1.css" />`

Likewise, add `.1` to any references in the copied templates to the installed image files contained in the `assets/images` directory.

Example: ``

Similarly, if you have modified any Web-container configuration settings that need to be carried forward, you must make corresponding changes manually in the new PingFederate deployment. For upgrading PingFederate version 6.9 and higher, you can simply copy over the relevant files from the Jetty configuration directory `<pf_install>/pingfederate/etc`. For existing versions prior to 6.9, first identify any changes made to the JBoss configuration, then make corresponding changes for the newer Jetty configuration.

For example, a commonly modified file prior to version 6.9 might be `jboss-service.xml` located in the directory `<pf_install>/pingfederate/server/default/deploy/jetty.sar/META-INF`. When changes are identified, make the same modifications at corresponding points in either the `jetty-admin.xml` or `jetty-runtime.xml` files located in the new Jetty configuration directory `<pf_install>/pingfederate/etc`.

Known Limitations

- As of version 6.5, PingFederate supports Thales nShield Connect HSM configurations. If your PingFederate installation is configured in a clustered environment with nShield Connect, you must copy the `<pf_install>/server/default/data/ncipher-kmdata-local` directory to the new installation manually and update the environmental variable `NFAST_KMLOCAL` to point to the new location. For more information, see *Additional Steps for Server Clusters* in the *Getting Started* guide.
- To avoid overwriting any PingFederate logging customization that may have occurred at your site, the Upgrade Utility determines whether `log4j.xml` has been modified in the existing installation. If so, new features in `log4j.xml` are *not* installed; the tool does not currently support automatic merging of customizations made to the existing logging configuration. (The XML file is located in the `<pf_install>/pingfederate/server/default/conf` directory.) Instead, if the existing file has changed since it was originally installed, the Upgrade Utility copies it to the new installation intact and renames the new configuration file using the current PingFederate version number.

The new configuration file contains several new logging capabilities added between version 6.0 and the current PingFederate version. (Please refer to product Release Notes for version 6.1+ and *Managing Log Files* in the *PingFederate Administrator's Manual*.) If the Upgrade Utility renames the new `log4j.xml` file, you must merge the two files if you want to retain previous logging changes and also use the new features. Or manually replicate any logging customizations in the new file and use it to replace the old one.

- As of PingFederate 6.2, TLS renegotiation was disabled by default for all HTTPS listeners/connectors. This security update affects only existing partner connections using WS-Trust STS and those making use of inbound mutual SSL/TLS authentication for the SOAP or artifact SAML bindings. In such cases, administrators must manually reconfigure `run.properties` to enable the secondary HTTPS port (see *Changing Configuration Parameters* in the *PingFederate Administrator's Manual*). In addition, affected connection partners must update their configurations to use the new endpoint.
- The Upgrade Utility does not migrate any data maintained in the PingFederate internal Hypersonic database or any external database. If, for example, SaaS Provisioning is enabled in the new PingFederate instance using the internal database, it is re-initialized from the provisioning source.
- The OGNL library used by PingFederate was upgraded with version 6.4. If you use OGNL expressions, we recommend retesting the expressions using the PingFederate administrative console.

PingFederate[®]

Version 7.3

Release Notes



© 2015 Ping Identity® Corporation. All rights reserved.

PingFederate 7.3 *Release Notes*
January, 2015

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
U.S.A.

Phone: 877.898.2905(+1 303.468.2882 outside North America)

Fax: 303.468.2909

Web Site: www.pingidentity.com

Trademarks

Ping Identity, the Ping Identity logo, PingFederate, PingAccess, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

Disclaimer

The information provided in this document is provided "as is" without warranty of any kind. Ping Identity disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Ping Identity or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Ping Identity or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.

Document Lifetime

Ping Identity may occasionally update online documentation between releases of the related software. Consequently, if this PDF was not downloaded recently, it may not contain the most up-to-date information. Please refer to documentation.pingidentity.com for the most current information.

From the Web site, you may also download and refresh this PDF if it has been updated, as indicated by a change in this date: **January 28, 2015**.

Contents

Release Notes Introduction	5
Enhancements for the 7.3 Release	5
Federation Hub	5
Administrative API Enhancements.....	5
Group Support for Inbound Provisioning.....	5
Support for wreply in WS-Federation SSO.....	6
Improved Redirect Validation	6
Security Enhancements	6
Improved Customizability	6
Other Enhancements	6
Upgrade Considerations.....	7
SSLv3 Disabled	7
New Representation for Multi-valued Attributes in WS-Federation Assertions	7
A New Database Table for OAuth Persistent Grant Extended Attributes	7
LDAP Adapter No Longer Supported.....	7
LDAP Filter Syntax Checking	7
HTML Form Adapter Enhancement.....	8
Requested (formerly SAML) AuthN Context Selector Process Order Changed	8
Multi-valued LDAP Attributes Passed to Outbound Provisioning OGNL Expressions.....	8
Cluster Bind Address Required	9
Decryption and Digital Signing Policy Changes	9
Key Transport Algorithm Deprecated.....	9
Deprecated Features.....	10
DSA Certificate Creation	10
JMX Monitoring Support for Outbound Provisioning.....	10
Known Issues.....	10
Complete Change List by Released Version.....	13
PingFederate 7.3 – January 2015.....	13
PingFederate 7.2 R2 – September 2014	13
PingFederate 7.2.1 – August 2014	14
PingFederate 7.2 – June 2014	14
PingFederate 7.1 R3 – March 2014.....	15
PingFederate 7.1 R2 – December 2013	16
PingFederate 7.1.4 – June 2014	16
PingFederate 7.1.3 – February 2014.....	17
PingFederate 7.1.2 – December 2013.....	17
PingFederate 7.1.1 – November 2013.....	17
PingFederate 7.1 – August 2013.....	17

PingFederate 7.0.1 – May 2013	18
PingFederate 7.0 – April 2013.....	18
PingFederate 6.11 – December 2012.....	19
PingFederate 6.10.1 – January 2013.....	20
PingFederate 6.10 – September 2012.....	20
PingFederate 6.9 – June 2012	20
PingFederate 6.8 – April 2012.....	20
PingFederate 6.7 – February 2012.....	20
PingFederate 6.6 – December 2011.....	21
PingFederate 6.5.2 – November 2011.....	21
PingFederate 6.5.1 – October 2011.....	21
PingFederate 6.5 – August 2011	21
PingFederate 6.5-Preview – April 2011	22
PingFederate 6.4.1 – February 2011	22
PingFederate 6.4 – December 2010.....	22
PingFederate 6.3 – August 2010	23
PingFederate 6.3-Preview – April 2010	23
PingFederate 6.2 – February 2010.....	23
PingFederate 6.1 – September 2009.....	24
PingFederate 6.1-Preview – June 2009.....	24
PingFederate 6.0 – March 2009	25
PingFederate 5.3 – December 2008.....	25
PingFederate 5.2 – August 2008	26
PingFederate 5.1.1 – July 2008.....	26
PingFederate 5.1 – April 2008.....	27
PingFederate 5.0.2 – March 2008	29
PingFederate 5.0.1 – January 2008.....	29
PingFederate 4.4.2 – October 2007.....	31
PingFederate 4.4.1 – June 2007	31
PingFederate 4.4 – May 2007	31
PingFederate 4.3 – March 2007	32
PingFederate 4.2 – December 2006.....	32
PingFederate 4.1 – October 2006.....	32
PingFederate 4.0 – June 2006	33
PingFederate 3.0.2 - February 2006.....	34
PingFederate 3.0.1 - December 2005.....	34
PingFederate 3.0 – November 2005.....	34
PingFederate 2.1 – July 2005.....	34
PingFederate 2.0 – February 2005.....	34

Release Notes Introduction

Welcome to PingFederate, Ping Identity's enterprise identity bridge. PingFederate enables outbound and inbound solutions for single sign-on (SSO), federated identity management, mobile identity security, API security, and social identity integration. Browser-based SSO extends employee, customer, and partner identities across domains without passwords, using only standard identity protocols (Security Assertion Markup Language—SAML, WS-Federation, WS-Trust, and OAuth).

These release notes summarize the changes in current and previous product updates.

Enhancements for the 7.3 Release

Note: For a condensed list of all enhancements for this and previous releases, see the [Complete Change List by Released Version](#) section, which also contains references to additional documentation.

Federation Hub

PingFederate now supports the ability to route SSO requests and responses between an IdP and an SP connection, potentially spanning disparate protocols, such as SAML 1.x, SAML 2.0, and WS-Federation. For example, an incoming SAML 2.0 IdP connection can be bridged to a WS-Federation SP connection. Additionally, Federation Hub features can centralize and simplify administrative overhead by multiplexing a single application to many IdP partners.

Many PingFederate feature areas have been enhanced to support Federation Hub use cases:

- Adapter Selectors are renamed as Authentication Selectors, and can now be used to choose amongst both IdP Connections and Adapters.
- “Map URLs to Adapter Instances” has been renamed as “Target URL Mapping.” It now appears under SP Application Integration Settings in the main Administrative Console page, and supports mapping a target URL pattern to either an SP Adapter or an SP Connection.
- A new SampleAuthenticationSelector SDK example has been created, which allows selection of an IdP Adapter or Connection, based on a user-supplied email address or domain.

Administrative API Enhancements

Creation, update and retrieval of SP Connections is now supported in the Administrative API. As with IdP Connections, SP Connection currently covers IdP and SP-initiated SAML 2.0 browser SSO profiles, including POST and redirect bindings.

Group Support for Inbound Provisioning

This release adds support for inbound provisioning of groups as defined in SCIM 1.1 (<http://www.simplecloud.info/>). This capability allows groups to be created, updated, and queried in a target identity store via SCIM calls to PingFederate. The provisioning target may be either an Active Directory data store or a custom Identity Store Provisioner plugin.

Support for wreply in WS-Federation SSO

This release includes support for the wreply parameter in WS-Federation SSO, which allows a single PingFederate SP connection to be used with multiple SharePoint 2013 hosted applications.

Improved Redirect Validation

Target Resource Validation has been renamed as Redirect Validation and moved to the central area of the Administrative Console, under Server Settings. Validation can now be applied to the TargetResource for SLO and InErrorResource parameters, in addition to TargetResource for SSO. Redirect validation is enabled by default in new installations of PingFederate. If upgrading from a previous version of PingFederate, it is recommended to enable the new validation options and add appropriate whitelist entries.

Security Enhancements

- For LDAP type data stores with LDAPS enabled, hostname verification of the certificate is now available. This option is enabled by default for all new data stores. When upgrading from a previous version of PingFederate, this option is disabled for existing data stores, but should be turned on for greater security.
- When using the anchored trust model for back-channel authentication or XML signature verification, it is now possible to restrict the certificate issuer DN.
- Elliptic Curve algorithms are now available for certificate creation and XML signatures. Additionally, RSA key based certificate creation has been enhanced to include RSA SHA-2 signing algorithms and 4096 bit keys.

Improved Customizability

- An Application Name and Application Icon URL can now be associated with SP Connections as well as Adapter-to-Adapter mappings. The new fields are accessible in the Adapter SDK.
- The response rendered by PingFederate for requests to the /pf/heartbeat.ping endpoint is now easily customizable via the heartbeat.page.template, and can be configured to include system information such as memory and CPU usage.
- A new template has been created, http.error.page.template.html, which allows customization of the page displayed to end users for standard HTTP errors, such as 404.
- A customizable favicon.ico now ships with PingFederate and applies to all end-user-facing endpoints.

Other Enhancements

- The LDAP Password Credential Validator can now be configured to return additional user attributes beyond the username and user DN. This enhancement simplifies attribute contract fulfillment in cases where additional data is required from the same LDAP directory store.
- The PingOne Directory Password Credential Validator is now bundled as part of the standard PingFederate installation.

- The HTML Form Adapter now supports a configurable maximum session lifetime.
- In Outbound Provisioning, PingFederate now supports provisioning nested LDAP groups.
- In Attribute Contract Fulfillment, PingFederate now supports retrieving nested LDAP groups.
- Upgraded JGroups to version 3.5.1.
- Upgraded Jetty to version 8.1.16.
- Over 60 other defects fixed.

Upgrade Considerations

Several specific modifications made since PingFederate 6.11 may affect existing deployments, as described in the following sections.

SSLv3 Disabled

To mitigate the POODLE attack, the SSLv3 protocol is disabled by default starting in PingFederate 7.3. It can be re-enabled by modifying the connector configuration in `jetty-runtime.xml` and `jetty-admin.xml`.

New Representation for Multi-valued Attributes in WS-Federation Assertions

Starting in PingFederate 7.3, multi-valued attributes in WS-Federation assertions are now represented as multiple `AttributeValue` elements under a single `Attribute` element. Previously, they were represented as a series of `Attribute` elements with the same name. The new behavior was implemented for compatibility with ADFS 2.0. To revert to the previous behavior, a setting is available in `wstrust-global-settings.xml`.

A New Database Table for OAuth Persistent Grant Extended Attributes

Starting in PingFederate 7.2 R2, a new database table needs to be created to support OAuth's 'Persistent Grant Extended Attributes'. The database script to create this table can be found in `<pf_install>/pingfederate/server/default/conf/access-grant/sql-scripts/access-grant-attribute-*.sql`.

LDAP Adapter No Longer Supported

Starting in PingFederate 7.2, the LDAP Java Adapter is no longer supported. This adapter was deprecated in PingFederate 6.6 and replaced by the LDAP Password Credential Validator (PCV), which can be used with the HTML Form or HTTP Basic IdP Adapters.

LDAP Filter Syntax Checking

Starting with PingFederate 7.2, LDAP filters only allow spaces in matched-against values.

Examples

`(|(sAMAccountName=${username})(employeeID=ID for ${username}))` is allowed; spaces in the matched-against value of “ID for \${username}” are valid.

`(|(sAMAccountName=${username})(employeeID=ID for ${username}))` is not allowed because this filter contain spaces outside of matched-against values.

Invalid filters cause SSO runtime failures. Error messages logged to `server.log` include:

```
Caused by: javax.naming.NamingException: [LDAP: error code 87 - Expected a closing parenthesis...
```

```
Caused by: javax.naming.NamingException: [LDAP: error code 87 - Unexpected closing parenthesis found...
```

We recommend reviewing LDAP filters and removing spaces outside of matched-against values after upgrade.

HTML Form Adapter Enhancement

Starting with version 7.1 R3, PingFederate tracks login attempts in the HTML Form Adapter. When the number of login failures reaches the Challenge Retries threshold defined in the adapter, the user is locked out for one minute. For more information, see *Administrator's Manual: HTML Form Adapter Configuration*.

Requested (formerly SAML) AuthN Context Selector Process Order Changed

In releases prior to 7.1 R2, when the Requested AuthN Context Adapter Selector received a list of authentication contexts, it used the last context that it could match, rather than the first. However, both the SAML and OpenID Connect specifications treat an authentication context list as appearing in order of preference. To align the Requested AuthN Context Adapter Selector with these specifications, the selection order was changed in 7.1 R2. With this release, the selector will use the first authentication context it can match, rather than the last.

Multi-valued LDAP Attributes Passed to Outbound Provisioning OGNL Expressions

In releases before version 7.1, if an OGNL expression was used to populate a SaaS-partner field in Outbound Provisioning, only the first value of a selected multi-valued LDAP attribute was used in the OGNL expression. As of PingFederate 7.1, this behavior was changed to use all values in the expression.

Note: If this new behavior conflicts with existing deployments, it may be reverted via the `supportMultiValuesFromDirectory` property located in

```
<pf_install>/pingfederate/server/default/data/config-store/com.pingidentity.provisioner.mapping.OgnlFieldMapper.xml
```

Cluster Bind Address Required

Starting with PingFederate 6.11, the `pf.cluster.bind.address` property (located in `<pf_install>/pingfederate/bin/run.properties`) is required when running PingFederate in a cluster.

Decryption and Digital Signing Policy Changes

Potential security vulnerabilities have resulted in the following changes to PingFederate as of version 6.11. In some cases, these may impact interoperability with partners:

- When acting as an SP and using the POST binding, PingFederate decrypts an assertion only when the SAML response has been signed. An unsigned SAML response that contains an encrypted assertion is rejected.

Note: Although strongly discouraged, this policy change may be reverted on a per-connection basis via the `EntityIdsToAllowAssertionDecryptionWithoutResponseSignature` property located in `<pf_install>/pingfederate/server/default/data/config-store/org.sourceid.saml20.profiles.sp.HandleAuthnResponse.xml`

- When acting as an IdP, PingFederate always signs a SAML response (even when the assertion is also signed) if it contains an encrypted assertion.

Note: Although strongly discouraged, this policy change may be reverted on a per-connection basis via the `EntityIdsToOmitResponseSignatureOnSignedEncryptedAssertion` property located in `<pf_install>/pingfederate/server/default/data/config-store/org.sourceid.saml20.profiles.idp.HandleAuthnRequest.xml`

- When acting as an IdP, PingFederate decrypts an encrypted NameID in an Attribute Query only when the request has been signed or the client has authenticated with basic or mutual TLS.

Key Transport Algorithm Deprecated

Due to security risks associated with the RSA-v1.5 algorithm used for key transport, it is no longer available for new connections. Existing connections in which this algorithm is configured continue to support it. However, we recommend upgrading such connections to use the newer algorithm RSA-OAEP (see *Administrator's Manual: Selecting an Encryption Certificate (SAML) and Choosing an Encryption Certificate*).

Note: The JCE provider libraries distributed with Luna and nShield Connect HSMs do not support the RSA-OAEP algorithm at the time of this release (7.1 R2). As a result, RSA-v1.5 is still allowed when deploying PingFederate with an HSM.

Deprecated Features

DSA Certificate Creation

As of PingFederate 7.3, it is no longer possible to create DSA key pairs in the certificate management pages of PingFederate. Import of DSA key pairs continues to be supported.

JMX Monitoring Support for Outbound Provisioning

As of PingFederate 7.2, the JMX Monitoring support for Outbound Provisioning has been deprecated and replaced with an audit file approach that is more inline with other auditing services offered by PingFederate.

JMX Monitoring for Outbound Provisioning can be re-enabled in 7.2 but will be removed in a future release.

Known Issues

- When using PingFederate with the Thales nShield Connect HSM, it is not possible to use an Elliptic Curve certificate as an SSL server certificate.
- If a source attribute has been configured for masking in an IdP adapter or IdP connection and the source attribute is mapped to OAuth's persistent grant USER_KEY attribute, then the USER_KEY attribute will not be masked in the server logs. Other persistent grant attributes will be masked.
- PingFederate's API Docs is not fully supported in IE9. The response headers will not show up in the API Docs' sandbox, due to limited Cross-origin resource sharing (CORS) support in IE9.
- The administrative API connection support is currently limited to IdP SAML 2 Browser SSO connections. As a result, it is possible to lose settings that are not supported by the administrative API (e.g. WS-Trust or SLO) when these settings are added through the UI and the connection is subsequently modified through the API.
- Concurrent API requests to the administrative API are currently not supported and may result in an unstable system.
- If an IdP connection is configured for multiple virtual server IDs, PingFederate will always use the default virtual server ID for IdP Discovery during an SP-initiated SSO event.
- OpenID Connect single-logout is not supported when the IdP session is created through OAuth attribute mapping.
- The RADIUS NAS-IP-Address is only included in Access-Request packets when the *pf.bind.engine.address* is set to an IPv4 address. IPv6 is not currently supported.
- The **Save Draft** feature is not available for connections that override plug-in settings from within the connection. In this case, the following DEBUG log message may be generated:

```
DEBUG [DraftManagerImpl] There was a NotSerializableException trying to serialize the draft
```

This message can be safely ignored.

- Adapter instances specified as “Sufficient” in a Composite Adapter configuration should be limited to adapter types that explicitly return control to PingFederate after a failure. Otherwise, the next adapter instance in an authentication “chaining” sequence (if any) may not be tried, and other unexpected behavior may occur.

The following adapters work correctly under the Sufficient authentication policy in failure mode:

- X509
- LDAP (legacy adapter)
- OpenID - Generic
- OpenID - Google
- IWA – returns control to PingFederate only if the failure is the result of invalid credentials after the configured number of retries
- HTML Form
- HTTP Basic

Note: This list is updated as other adapters are modified, tested, and released.

- The anchored-certificate trust model cannot be used with the single logout (SLO) redirect binding since the certificate cannot be included with the logout request.
- PingFederate cannot simultaneously log the audit log to multiple databases and/or ArcSight CEF syslog. The audit log can only use a single log4j appender. See the `log4j.xml` file in `<pf_install/pingfederate/server/default/conf>` for additional details.
- For a scenario involving SP-initiated SLO with multiple SPs in which the initiating SP is using a SOAP binding and the other SPs are using one of the front-channel bindings (Artifact, Redirect, or POST) along with a front-channel adapter within the IdP, logout with the front-channel adapter fails. When logout fails with the adapter (a technical limitation, since with SOAP-based SLO, the server does not have access to the browser to kill a session established with a front-channel adapter), any other IdP adapters that are configured for the connection, and for which logout needs to occur will not be invoked for logout. This includes back-channel (e.g., SOAP-based) adapters.
- Using the browser’s navigation mechanisms (e.g., the **Back** button) causes inconsistent behavior in the administrative console. Use the navigation buttons provided at the bottom of screens in the PingFederate console.
- If authenticated to the PingFederate administrative console using certificate authentication, a session that has timed out may not appear to behave as expected. Normally (when using password authentication), when a session has timed out and a user attempts some action in the console, the browser is redirected to the login page and then to the Main Menu once authentication is complete. Similar behavior applies for certificate authentication, in principle. However, since the browser may automatically resubmit the certificate for authentication, the browser may redirect to the Main Menu and not the login page.
- LDAP referrals return an error and cause provisioning to fail if the User or Group objects are defined at the DC level, and not within an OU or within the Users CN.
- When an nShield Connect HSM in a HA nShield Connect cluster is shutdown, users will receive exceptions in the console when trying to create private keys for both digital signatures and SSL.

Before creating new private keys, the nShield Connect HSM should be restored to a normal state (up on the network up, HSM up with OCS cards in their slots), and PingFederate should be restarted.

- PingFederate can enforce the masking of sensitive attribute values only within its own code base. External code such as adapter implementations and other product extensions may log attribute values in the clear even when they have been designated to be masked in the GUI. If sensitive attribute values are a concern when using such components, the logging level for the specific component can be adjusted in the `log4j.xml` file to the appropriate threshold to prevent attribute values from appearing in log files.
- When PingFederate is acting as a WS-Trust STS, if it receives a request on the STS endpoint with the namespace element set to an invalid value of `http://schemas.xmlsoap.org/ws/2005/02/trust/` (i.e., with a trailing slash), it does not normalize this to the valid namespace of `http://schemas.xmlsoap.org/ws/2005/02/trust` (i.e., without the trailing slash) and fails the transaction. In this case, the workaround is to have the client set the namespace element to the valid namespace of `http://schemas.xmlsoap.org/ws/2005/02/trust`.
- When upgrading to PingFederate 6.8 or higher, all persistent grants for any existing OAuth deployments using a MySQL database will expire. To address this issue, the `expires` column in the `pingfederate_access_grant` table should be nulled prior to the upgrade. If necessary, contact Ping Identity support for assistance.

Note: The remaining items in this list concern limitations that apply to the use of the PingFederate configuration-migration scripting tool, `configcopy`.

- If you are using `configcopy` to copy all connections, channels, data sources, adapters, or token translators and you choose to set override properties, the override is applied to all instances. It is recommended that you use care when applying overrides for copy-all operations.
- The `configcopy` tool supports copying only a single reference for each of the following that are defined for a given connection: adapter, data source, Assertion Consumer Service URL, Single Logout Service URL, and Artifact Resolution Service URL. If you have multiple adapters, data stores, or any of the aforementioned service URLs associated with a given connection, only the first reference to each is copied.
- The `configcopy` tool does not support creation of configuration data that does not exist in the source. If you choose to set an override parameter for a parameter that does not exist in the source configuration, the behavior of the target system is not guaranteed.
- The `configcopy` tool, when used for copying plug-in configurations (including adapters, token translators, and custom data stores), does not currently support overrides of complex data structures, including tables, extended contract attributes, and masked fields.
- When using `configcopy` to copy connection data, any SOAP SLO endpoints defined in the source are not copied to the target, even if the SOAP SLO endpoint is the only SLO endpoint defined at the source. These must be manually added to the target.

Complete Change List by Released Version

PingFederate 7.3 – January 2015

- Federation Hub
- SampleAuthenticationSelector SDK Sample
- SP Connections API
- Group Support for Inbound Provisioning
- Support for wreply in WS-Federation SSO
- Improved Redirect Validation
- Hostname verification for LDAPS data stores
- Issuer restriction in anchored trust model for back-channel authentication and XML signature verification.
- Elliptic Curve algorithms for certificate creation and XML signatures.
- RSA certificate creation enhanced to include RSA SHA-2 signing algorithms and 4096 bit keys.
- Application Name and Application Icon URL for SP Connections and Adapter-to-Adapter mappings.
- Customizable response from /pf/heartbeat.ping.
- Customizable template for HTTP error pages.
- Customizable favicon.ico
- LDAP Password Credential Validator can now be configured to return additional user attributes beyond the username and user DN.
- PingOne Directory Password Credential Validator is now bundled as part of the standard PingFederate installation.
- Key algorithm and key size are now displayed in certificate management pages and the certificate details popup.
- HTML Form Adapter now supports a configurable maximum session lifetime.
- In Outbound Provisioning, PingFederate now supports provisioning nested LDAP groups.
- In Attribute Contract Fulfillment, PingFederate now supports retrieving nested LDAP groups.
- Upgraded JGroups to version 3.5.1.
- Upgraded Jetty to version 8.1.16.
- Over 60 other product issues resolved.

PingFederate 7.2 R2 – September 2014

- Multiple Access Token Management Plug-in Instances
- Improved Attribute Mapping for OAuth Access Token Contract

- Multiple Data Source Lookup for OAuth Workflows
- OpenID Connect / OAuth 2.0 Form POST Response Mode
- Include User Info in the OpenID Connect ID Token
- Administrative API Enhancements
 - Configuration Archive Management
 - IdP Connection Metadata Export / Import
 - Certificate Management of Trusted CAs, SSL Client Keys, and SSL Server Certificates
 - Data Sources
 - Password Credential Validators
 - IdP-to-SP Adapter Mapping
- Administrative Console Enhancements
- Other Enhancements
 - The OAuth `client_id` parameter is now available in the IdP Adapter SDK interface
 - Support SID encoding for binary LDAP attributes (enabling claims-based authentication to Microsoft Outlook Web Access)
 - Various security enhancements
 - Over 40 other product issues resolved

PingFederate 7.2.1 – August 2014

- Fixed an issue for Office 365 Outlook use cases where user accounts could be locked after changing their passwords in Active Directory.
- Fixed an issue where users would not be provisioned correctly under certain conditions.
- Fixed an issue that prevented additional "Actions" in adapter configuration from executing correctly.

PingFederate 7.2 – June 2014

- Multiple Virtual Server IDs
- OpenID Connect-based Centralized Session Management
- LDAP Enhancements
 - Improved performance on LDAP bind operations
 - Ability to control timeout on LDAP attribute lookups
 - Better connection cleanup when an idle LDAP connection is removed from the pool
- Administrative API Enhancements
 - IdP and SP Adapters
 - Access Token Management (OAuth)

- Custom HTTP Response Headers
- Improved Target Resource Validation
- Outbound Provisioning Monitoring
- New Username Token Processor
- Redesigned Default User-Facing Screens
- Other Enhancements
 - Support for Java 8
 - Support for OAuth Symmetric Proof of Possession for Code Extension
 - Upgraded JGroups to 3.4.4.Final
 - Various security enhancements
 - Over 50 other product issues resolved

PingFederate 7.1 R3 – March 2014

- Administrative API Enhancements
 - Server Settings (OAuth Use Cases)
 - OAuth Settings
 - Authorization Server Settings
 - OpenID Connect Policy Management
 - Client Management
 - IdP Adapter Mapping
 - Access Token Mapping
- OAuth Enhancements
 - OAuth 2.0 Token Revocation (RFC 7009)
 - Access Grant API
- RADIUS Support Extended
 - Challenge-Handshake Authentication Protocol (CHAP) Support
 - Vendor-Specific Attributes
 - NAS-IP-Address Passed in Access-Request
- Enhanced Support for Reverse Proxy Deployments
- Other Enhancements
 - Allow PingFederate's session cookie to be optionally configured to be persistent
 - Allow the Access Token Verification/Validation Grant Type to be used in combination with other grant types
 - Various security enhancements

- Over 20 other product issues resolved

PingFederate 7.1 R2 – December 2013

- Administrative API (see *Administrator's Manual: PingFederate Administrative API*)
- Tracking ID Enhancements
- Outbound Provisioning
 - Microsoft SQL Server Support added for Internal Provisioning Data Store (see *Administrator's Manual: Configuring Outbound Provisioning Settings*)
 - OAuth 2.0 Bearer Token Authorization for SCIM 1.1 Plugin (see *Administrator's Manual: Defining a Provisioning Target*)
 - Ability to Define Deprovision Method for SCIM 1.1 Plugin (see *Administrator's Manual: Defining a Provisioning Target*)
- OAuth and OpenID Connect Enhancements
 - Extended OAuth to allow multiple scopes to be grouped together and referenced as a single scope, enabling scope downgrade during token refresh
 - Allow administrators to define a maximum token lifetime for OAuth Access Tokens
 - OpenID Connect now supports requesting that the Authorization Server use specified authentication contexts when processing the authentication request
- Added TemplateRendererUtil class and template-render-adapter-example to PingFederate SDK
- Allow administrators to define a maximum token lifetime for OAuth Access Tokens
- Extended HTML Form Adapter to allow usernames to be stored and pre-populated in the login form after a user's first successful authentication
- OpenID Connect now supports requesting that the Authorization Server use specified authentication contexts when processing the authentication request
- Partner Entity ID (Connection ID) now available as input parameter to all Identity Store Provisioner requests
- Extended OAuth to allow multiple scopes to be grouped together and referenced as a single scope, allowing scope downgrade during token refresh
- Reintroduced Luna HSM Support
- Upgraded Jetty to 8.1.14
- Upgraded JGroups to 3.3.4.Final
- Various security enhancements
- Over 40 other product issues resolved

PingFederate 7.1.4 – June 2014

- Addressed a potential CSRF issue with the PingFederate Administration console.

- Updated the default Multiple SSO in Progress template to escape additional input variables (`speed.bump.template.html`).
- Addressed an issue where Office 365 Outlook users can have their user accounts to be locked out after they change their Active Directory passwords.
- Corrected the Content-Type header for interoperability with Office 365 OneDrive.
- Made available the `$HttpServletRequest` object in the IdP logout template (`template.idp.logout.success.page.template.html`).
- Resolve an issue with body-less HTML Form POST messages sent by Internet Explorer when HTML Form based authentication follows an unsuccessful IWA authentication within a Composite Adapter.
- Made an improvement to correctly handle LDAP queries for Distinguished Names having forward slashes (/).

PingFederate 7.1.3 – February 2014

- Corrected potential security vulnerability.

PingFederate 7.1.2 – December 2013

- Corrected an issue with artifact binding in clustered deployments that resulted in sporadic failed transactions. This issue only applied to version 7.1.1.

PingFederate 7.1.1 – November 2013

- Corrected potential security vulnerability
- Fixed XML namespace issue that resulted in failed SSO transactions at partners with hardcoded dependencies on the `samlp` prefix
- Resolved XML parsing issue that resulted in PingFederate rejecting incoming authentication requests
- Corrected logging issue that resulted in AJP-based transactions to fail

PingFederate 7.1 – August 2013

- Added support for Outbound Provisioning via PingOne (see *Administrator's Manual: Configuring Outbound Provisioning*)
- Added identity store SDK for Inbound Provisioning (see *Administrator's Manual: Configuring Inbound Provisioning*)
- Provided Remote Authentication Dial-In User Service (RADIUS) Support for console logon and password credential validation (see *Administrator's Manual: Alternative Console Authentication, Validating Password Credentials*)
- Added hierarchical (parent/child) configurations for plug-in instances (see *Administrator's Manual: Configuring Inbound Provisioning*)

- Related plug-in instance override capability added for connections (see *Administrator's Manual: Selecting an Adapter Instance, Choosing an Adapter Instance*)
- Enabled overriding the global Default Target URL for connections and adapter-to-adapter mapping (see *Administrator's Manual: SSO configuration sections for IdP and SP connections and for IdP-to-SP adapter mapping*)
- Made STS client authentication details available for token authorization (see *Administrator's Manual: WS-Trust STS Configuration*)
- Extended JMX runtime monitoring support (see *Administrator's Manual: Runtime Monitoring Using JMX*)
- Enabled usage checking for certificates, adapters, token translators, credential validators and data stores (see *Administrator's Manual: relevant configuration sections*)
- Store OAuth client configuration as XML files (new default, database storage optional) (see *Administrator's Manual: Configuring Inbound Provisioning*)
- Upgraded Jetty to 8.1.9
- Upgraded JGroups to 3.3.0.Final
- Various security enhancements
- Over 80 other product issues resolved

PingFederate 7.0.1 – May 2013

Addressed potential security vulnerabilities found since the PingFederate 7.0 initial, limited-availability release.

PingFederate 7.0 – April 2013

- Added support for System for Cross-domain Identity Management (SCIM)
 - Inbound Provisioning (see *Administrator's Manual: Configuring Inbound Provisioning*)
 - Outbound Provisioning (see *Administrator's Manual: Configuring Outbound Provisioning*)
- Initial Support for OpenID Connect
- New Context sources available in Token Authorization and Attribute Mapping workflows
 - HTTP Request
 - *Client IP* Request
- Administrative Console Enhancements
- Upgraded Jetty to 8.1.8
- Added support for the SAML AuthnRequest Scoping element Request (see *Administrator's Manual: Configuring the Requested AuthN Context Selector*)
- Created the Cluster Node Adapter Selector for use within Adapter Selection Request (see *Administrator's Manual: Configuring the Cluster Node Selector*)

- Allow PingFederate runtime context path to be customized Request (see *Administrator's Manual: System Administration*)
- Security Enhancements
- Over 50 product issues resolved

PingFederate 6.11 – December 2012

- Added password update via HTML Form IdP Adapter (see *Administrator's Manual: Configuring the HTML Form IdP Adapter*)
- Enhanced IdP Adapter Selector functionality:
 - Decision Tree Processing (see *Administrator's Manual: Mapping Selector Results to Adapter Instances*)
 - Connection Set Selector (see *Administrator's Manual: Configuring the Connection Set Selector*)
 - HTTP Header Selector (see *Administrator's Manual: Configuring the HTTP Header Selector*)
 - OAuth Scope Selector (see *Administrator's Manual: Configuring the OAuth Scope Selector*)
- Added localization for user-facing Web pages (see *Administrator's Manual: Localization*)
- Added capability of defining globally an HTTP Header containing client IP addresses (see *Administrator's Manual: Defining an HTTP Header for Client IP Addresses*)
- Added OAuth fine-grain scope handling and JWT access-token support (see *Administrator's Manual: Configuring JSON-Token Management*)
- Added Dynamic JVM Resource Allocation
- Upgraded JGroups to 3.3.0.Alpha1 (snapshot from 10/30/12)
- Extended the Consent Form template to include the \$entityId parameter
- Extended the OAuth Grants Management template to show grant type, scope, and the date/time a grant was issued and updated
- Improved SLO handling in scenarios where a user performs multiple SSO requests
- Improved the ability for administrators to define validation rules for HTTP request parameters
- Improved JDBC data store attribute lookup functionality to allow for retrieval of multiple values from a single database column (see *Administrator's Manual: Configuring a JDBC Database Connection*)
- Made security enhancements
- Removed the requirement to include TokenProcessorId/TokenGeneratorId query parameters for STS requests to an IdP/SP connection when only a single instance of the token-processor/generator type is configured for the connection
- Enabled PingFederate to use Jetty's NIO (New I/O) backed SSL Connectors by default
- Enhanced Microsoft Office 365 support to allow for Kerberos interoperability with active clients built on top of Microsoft Online Services Sign-In Assistant

- Enhanced LDAP error code handling in the LDAP Username Password Credential Validator to enable more specific error messages to be provided to end users
- Enabled PingFederate to interoperate with Microsoft Dynamics CRM 2011

PingFederate 6.10.1 – January 2013

- Replaced OpenToken adapter with version 2.5.1 to capture security enhancements
- Other changes to address potential security vulnerabilities

PingFederate 6.10 – September 2012

- Token Authorization (see *Administrator's Manual: About Token Authorization*)
- STS token exchange mapping (see *Administrator's Manual: STS Token Exchange Mapping*)
- OAuth client mutual TLS authentication (see *Administrator's Manual: Configuring a Client*)
- OAuth 2.0 final draft compliance

PingFederate 6.9 – June 2012

- Microsoft Office 365 interoperability
- STS transaction events logged to audit log (see *Administrator's Manual: System Administration*)
- Upgraded Jetty and removed underlying JBoss infrastructure

PingFederate 6.8 – April 2012

- Added centralized AD Domain/Kerberos Realm configuration (see *Administrator's Manual: Security Management*)
- Added OAuth Client Management REST API (see *Administrator's Manual: Web Service Interfaces*)
- Added optional expiration of OAuth persistent grants (see *Administrator's Manual: OAuth Configuration*)
- Added multiple redirect URIs per OAuth client (see *Administrator's Manual: OAuth Configuration*)
- Added optional restricted scope subsets per OAuth client (see *Administrator's Manual: OAuth Configuration*)
- Added configurable consent page omission per OAuth client (see *Administrator's Manual: OAuth Configuration*)
- Added OAuth transaction events logged to audit log (see *Administrator's Manual: System Administration*)

PingFederate 6.7 – February 2012

- Added Splunk Application for PingFederate (see *Administrator's Manual: System Administration*)
- Improved administrative console navigation and save performance

- Added LDAP connection pooling options for LDAP data stores (see *Administrator's Manual: System Settings*)

PingFederate 6.6 – December 2011

- Added contextual IdP Adapter selection using Adapter Selectors (see *Administrator's Manual: Key Concepts*)
- Added ability to chain multiple IdP adapters together using the Composite Adapter (see *Administrator's Manual: Key Concepts*)
- Added the ability to use multiple IdP data stores for attribute retrieval and mapping into an IdP attribute contract (see *Administrator's Manual: Key Concepts*)
- Added an HTML Form adapter and HTTP Basic adapter to replace the LDAP Authentication adapter (see *Administrator's Manual: Key Concepts*)
- Support for the OAuth SAML 2.0 Bearer Assertion Grant Type (see *Administrator's Manual: OAuth Configuration*)
- Added an Admin Console Help system updater (see *Administrator's Manual: System Settings*)
- Added IPv6 support

PingFederate 6.5.2 – November 2011

Security update since the PingFederate 6.5.1 release

PingFederate 6.5.1 – October 2011

Security updates since the PingFederate 6.5 release

PingFederate 6.5 – August 2011

Note: The PingFederate 6.5 release includes the features described below as well features that were added in a limited-distribution "Preview" release, described in the next section.

- PingFederate now functions as an OAuth 2.0 Authorization Server (see *Administrator's Manual: OAuth Configuration*)
- Added support for Thales (nCipher) nShield Connect HSM (see *Getting Started: Using the Thales nShield Connect HSM*)
- Account Linking can use an LDAP directory for a persistent data store in addition to a relational database system (see *Administrator's Manual: System Settings*)
- User-Defined Attribute Namespaces can be specified for Browser SSO protocols (similar to what was added to WS-Trust STS) to allow for better Microsoft interoperability (see *Administrator's Manual: Key Concepts*)
- Adapter to Adapter mapping now counts as a licensed connection (see *Administrator's Manual: System Settings*)

- LDAP Adapter updated to 2.2 with new default Web form login template (see *Administrator's Manual: LDAP Adapter Configuration*)
- Jetty version upgrade from 6.1.7 to 6.1.26

PingFederate 6.5-Preview – April 2011

- Full STS metadata Claims Provider and Relying Party interoperability with Microsoft WIF, WCF, and ADFS 2.0
- Support multiple token-processor instances of the same token type (see *Administrator's Manual: IdP Configuration for STS*)
- Added SAML HoK subject confirmation in the SAML Token Generator (see *Administrator's Manual: SP Configuration for STS*)
- Added option for STS SAML token KeyInfo to use a signing certificate reference rather than the full signing certificate (see *Administrator's Manual: IdP Configuration for STS*)
- Session-state modifications to support simultaneous and nested SSO transactions
- IdP adapter session handling for IdP adapters that rely on PingFederate for session management to allow for consecutive requests without prompting for credentials

PingFederate 6.4.1 – February 2011

- Corrected license expiration date calculation

PingFederate 6.4 – December 2010

- Support standard .NET WS-Trust Federation Bindings (see *Administrator's Manual: Key Concepts*)
- Support SAML 2.0 token Holder of Key (HoK) subject confirmation (see *Administrator's Manual: WS-Trust STS Configuration*)
- Added Metadata Exchange (MEX) endpoint for WIF client to generate bindings automatically for Username, X.509, and SAML tokens (see *Administrator's Manual: Application Endpoints*)
- Added support for WS-Trust 1.4 ActAs property
- Added two-factor authentication capability with the VeriSign® Identity Protection (VIP) Authentication Service Adapter (see *Administrator's Manual: Identity Provider SSO Configuration*)
- OpenToken Adapter 2.4.1 updated to correct issue with Cookie Transport Method and Replay Prevention
- Expanded digital signature secure hash algorithm types - SHA1, SHA256, SHA 384, and SHA512 (see *Administrator's Manual: sections covering applicable certificate-selection screens*)
- The provisioning log can be written to a database—Oracle, Microsoft SQL Server, and MySQL databases supported (see *Administrator's Manual: System Administration*)
- Added SAML protocol support for AuthnContextDeclRefs (see *Administrator's Manual: Application Endpoints*)

PingFederate 6.3 – August 2010

Note: The PingFederate 6.3 release includes the features described below as well features that were added in a limited-distribution “Preview” release, described in the next section.

- PingFederate STS claims-based identity capabilities extended to support interoperability with Microsoft WIF and WCF client frameworks (see *Administrator’s Manual: Key Concepts*)
- Expanded SNMP monitoring variables available in the management information base (MIB) (see *Administrator’s Manual: System Settings*)
- Increase the default PingFederate HTTP header buffer size to 8k
- Key stores and key store passwords are dynamically generated per installation
- The default SSL server certificate is generated upon initial startup if an SSL certificate does not exist
- LDAPS trust configuration no longer requires a server restart to take effect

PingFederate 6.3-Preview – April 2010

- Added support for logging to the ArcSight Common Event Format (CEF) (see *Administrator’s Manual: System Administration*)
- Added ability to log to a database with failover to file—Oracle, SQL Server, and MySQL databases supported (see *Administrator’s Manual: System Administration*)
- Added ability to disable automatic multi-connection validation if the validation time is causing excessive delay (see *Administrator’s Manual: System Settings*)
- Extended JDBC Express Provisioning to support MS SQL Server stored procedures (see *Administrator’s Manual: Service Provider SSO Configuration*)
- Added replay prevention capability to the OpenToken IdP Adapter bundled with PingFederate

PingFederate 6.2 – February 2010

- Added IdP-to-SP adapter mapping, which allows user attributes from an IdP adapter to be directly mapped to an SP adapter on the same PingFederate server to create an authenticated session or security context, without the need to generate SAML messages in between (see *Administrator’s Manual: System Settings*)
- Provides enhanced logging capabilities including a new audit log, logfilter utility, and ability to log to any accessible file-server directory (see *Administrator’s Manual: System Administration*)
- Provides enhanced support for configuration automation including certificate and key management, configuration archive management, and ancillary deployment files (see *Administrator’s Manual: System Administration*)
- Extended JDBC Express Provisioning to support MS SQL Server Identity column types (see *Administrator’s Manual: Service Provider SSO Configuration*)
- Added a Logout Endpoint to the LDAP Authentication Adapter (see *Administrator’s Manual: LDAP Adapter Configuration*)

- In clustered mode, the default Inter-Request State Management methodology is now group RPC-based instead of cookie-based (see the Server Clustering Guide)
- Added ability to extract CN from DN and extract username from email address for provisioner attributes (see *Administrator's Manual: Identity Provider SSO Configuration*)
- In Luna HSM mode, added the ability to specify the location to store Trusted CA certificates, either in the Sun Java key store or the Luna HSM (see the configuration file `org.sourceid.config.CoreConfig.xml` in the `pingfederate/data/config-store` directory)

PingFederate 6.1 – September 2009

Note: The PingFederate 6.1 release includes the features described below as well features that were added in a limited-distribution “Preview” release, described in the next section.

- Provides support for simplified PingFederate Express connection configuration and export (see *Administrator's Manual: Identity Provider SSO Configuration*)
- Extends support for configuration automation, including listing, copying, and updating features for SaaS Provisioning channels and for Token Translators (see *Administrator's Manual: System Administration*)
- Provides enhanced support for SaaS Provisioning Health and Status Monitoring via JMX (see *Administrator's Manual: System Settings*)
- Provides licensing enhancements including support for organizational licenses, licenses that contain international characters, and Web based license import (see *Administrator's Manual: System Administration*)
- Enhances the trust model to include support for anchored certificates, which allows certificates to be included in federation-transaction messaging and used for signature verification if, the given certificate matches the registered Subject DN and is issued by a certificate authority registered as a Trusted CA with PingFederate (see *Administrator's Manual: Key Concepts*)
- Supports “SP Lite”, “IdP Lite”, and “e-Gov” Liberty Interoperability profiles for SAML 2.0

PingFederate 6.1-Preview – June 2009

- Provides support for Express Provisioning to a JDBC data source (see *Administrator's Manual: Service Provider SSO Configuration*)
- Extends support for configuration automation, including listing, copy and update features for data stores and server settings (see *Administrator's Manual: System Administration*)
- Supports certificate-based authentication to the PingFederate administrative console (see *Administrator's Manual: System Administration*)
- Added UI-based data-archive deployment, as well as better error handling for common errors encountered (see *Administrator's Manual: System Administration*)
- Supports mapping of attributes passed in via a WS-Trust STS request to the outgoing token (see *Administrator's Manual: WS-Trust STS Configuration*)

- Supports logging of a transaction ID associated with every log entry for a given request to the PingFederate server
- Corrects defects reported by customers in the previous release

PingFederate 6.0 – March 2009

- PingFederate now includes a WS-Trust Security Token Service (STS), enabling organizations to extend identity management to Web Services. The PingFederate STS shares the core functionality of PingFederate, including console administration, identity and attribute mapping, and certificate security management (see *Getting Started: WS-Trust STS Configuration*).
- PingFederate extends support for configuration automation, including connection management and adapter management via the existing command-line tool (see *Getting Started: Installation*).
- PingFederate supports enhanced connection based licensing capabilities.
- PingFederate provides transaction based licensing capabilities for evaluation phase license enforcement.
- PingFederate allows administrators to specify the use of a separate certificate that is used for access to the administrative console and a different certificate for runtime processing. (see *Administrator's Manual: System Settings*).
- PingFederate supports configuration of LDAP Groups who are allowed access to the PingFederate Admin application based on PingFederate defined roles (see *Administrator's Manual: System Administration*).
- PingFederate supports definition of LDAP data stores such that the connection URI for multiple LDAP servers can be specified as the connection string for that LDAP data store (see *Administrator's Manual: System Settings*).
- PingFederate supports the Virtual List View (VLV) paging mechanism for retrieval of subsets of large result sets returned from the source LDAP data store during the provisioning process. This can significantly enhance performance for retrieval of data from Sun Directory Server (SDS) and similar LDAP servers that support VLV paging.
- PingFederate now stores the PingFederate software version within the configuration store.
- A number of defects reported by customers that existed in the previous release of PingFederate were addressed.

PingFederate 5.3 – December 2008

- PingFederate can be run as a service on Windows 64-bit platforms in addition to Windows 32-bit platforms and Linux platforms (see *Getting Started: Installation*).
- PingFederate now supports deployment of SaaS Provisioning plug-ins (JAR files) via a separate installation package (documented in Connector packages).
- PingFederate now supports automating configuration via a command line utility for connection management (see *Administrator's Manual: System Administration*).

- PingFederate now supports capabilities for monitoring and control of the SaaS Provisioning configuration and data via a command line tool (see *Administrator's Manual: System Administration*).
- PingFederate now supports validation of certificate revocation information via OCSP (see *Administrator's Manual: System Settings*).
- PingFederate can now be deployed on Java 6 (JDK 1.6) platforms.
- PingFederate supports access to additional parameters on both the IdP side and the SP side via OGNL expressions (see *Administrator's Manual: Identity Provider SSO Configuration and Service Provider SSO Configuration*).
- PingFederate supports “SP Lite” and “IdP Lite” Liberty Interoperability profiles for SAML 2.0.
- PingFederate supports configuration of the name, domain, and path for the cookie used for conveying state information between servers when cookie-based clustering has been configured.
- A number of past Known Issues and Limitations were addressed.

PingFederate 5.2 – August 2008

- A PingFederate IdP server now provides support for provisioning to selected SaaS providers. PingFederate supports provisioning of user account data from LDAP directories including Active Directory and Sun Directory Server. PingFederate stores synchronization data in JDBC data stores including Hypersonic (for demonstration purposes) and Oracle.
- PingFederate supports quick-connection templates to selected SaaS Providers, including Google Apps, and Salesforce.com

PingFederate 5.1.1 – July 2008

This release corrected several issues, including:

- SP signature verification was failing for assertions containing UTF-8 characters.
- In Windows the PingFederate server was unable to start when the JAVA_HOME system variable contained a space.
- Versions of the OpenToken library were placed in the wrong directory.
- Single Logout (SLO) with two SPs was not being performed for the IdP session(s).
- For SLO with three or more SPs, SP sessions were being stranded.
- Specific to PingFederate 5.1, Custom Data Sources no longer could be used for Adapter Contract fulfillment.
- When testing certain types of OGNL expressions, important error details were being lost when evaluation of these expressions failed.
- For SLO with at least two SPs, under certain circumstances error messages from SPs that did not initiate the SLO were not being processed correctly by the IdP.
- A PingFederate SP instance, when used with the OpenToken adapter, was converting a plus “+” character to a space “ ” when constructing the URL for final redirect.

- Signature validation was failing within a PingFederate SP instance when it received an SLO message in which the SAML_SUBJECT was being encrypted.
- PingFederate 5.1 SP instance was no longer supported SiteMinder SSO Zones.

PingFederate 5.1 – April 2008

- The default behavior when PingFederate cannot access a Certificate Revocation List (CRL) is now set correctly. The server no longer treats a non-retrievable CRL as a reason to label certificates as revoked. CRL processing behavior is managed by the revocation-checking-config.xml file in the /pingfederate/server/default/data/config-store directory.
- The IP address to which PingFederate's SNMP agent binds is now controlled by the pf.monitor.bind.address property in the run.properties file (*Administrator's Manual: System Administration*).
- Building either of the two example adapters included in the PingFederate SDK no longer fails with an error regarding a missing README.txt.
- The PingFederate server now correctly maintains temporary files within the /pingfederate/server/default/tmp directory. The server no longer writes temporary files to the tmp directory of the user running the server.
- Express Provisioning allows user accounts to be created in an LDAP repository and updated directly by an SP PingFederate. User provisioning occurs as part of SSO processing and may be used with any IdP partner (*Administrator's Manual: Managing IdP Connections*).
- The Signature Policy screen in SAML IdP connections contains improved language clarifying how signatures are used to guarantee authenticity of SAML messages (*Administrator's Manual: Managing IdP Connections*).
- The PingFederate package now contains v6.1.7 of Jetty. Jetty is the servlet container used by PingFederate.
- The PingFederate SDK contains a ConfigurationListener interface that may be utilized by developers building adapters. This interface contains methods invoked by the server in response to certain adapter-instance lifecycle events such as creation and deletion.
- Adapter-instance Summary screens now display adapter-instance configuration values specified within a TableDescriptor.
- After the third consecutive failed login attempt, an administrator is blocked from accessing the administrative console for a configurable amount of time (default = 60 seconds).
- When changing an administrator password, the server now forces the new password to differ from the existing password (*Administrator's Manual: System Administration*).
- Access to services exposed by the PingFederate server now requires client authentication. These services include Attribute Query, JMX, and Connection Management. An administrator may choose to require client authentication for access to the SSO Directory Service. An ID and Shared Secret comprise the credentials needed for authentication (*Administrator's Manual: Security Management; Administrator's Manual: Web Service Interfaces*).
- For security, the use of "Expression" in contract fulfillment screens is now disallowed by default. For backward compatibility, customers deploying a configuration archive from a previous version of PingFederate in which expressions were used will continue to have access to expressions. Allowing

expressions creates a potential security concern in the PingFederate administrative console. (*Administrator's Manual: Using Attribute Mapping Expressions*)

- The Quick-Start SP Application no longer uses an OGNL expression in fulfilling the SP adapter contract.
- HTTP TRACE requests sent to PingFederate now result in an HTTP 403 Forbidden response.
- By default, “weak” ciphers are no longer supported during SSL handshaking. (For more information as to which cipher suites the server supports, examine the `com.pingidentity.crypto.SunJCEManager.xml` file in `pingfederate/server/default/data/config-store`.)
- It is no longer possible for an administrator to circumvent role and access permissions within the administrative console by direct URL access. The server evaluates HTTP requests for a URL against an administrator’s assigned role(s) and responds appropriately.
- The PingFederate runtime server’s HTTP listener is now turned off by default. Only messages sent over HTTPS are accepted. This may be controlled in the `run.properties` file in the `pingfederate/bin/directory`.
- Use of class and package names specific to a PingFederate version were removed from sample source code contained in the PingFederate SDK.
- The `/idp/startSSO.ping` endpoint now supports an optional ACSIdx query parameter for SAML v2 partners. When provided, the PingFederate IdP attempts to send the SAML Assertion to the Assertion Consumer Service corresponding to the specified Index (*Administrator's Manual: Application Endpoints*).
- The initial and maximum JVM heap sizes are set to 256 MB and 1024 MB, respectively, by default. These changes should improve runtime performance on servers with sufficient memory. These settings reside in the `run.bat` and `run.sh` files of the `pingfederate/bin/directory`.
- During server startup, PingFederate now reports relevant environment variables and adapter-instance information to the `server.log`.
- Existing partner connections can be deleted through a SOAP call from an external client application. (*Administrator's Manual: Web Service Interfaces*).
- The `pf-legacy-runtime.war` file is no longer deployed by default. This WAR file allows a PingFederate server to continue support of legacy endpoints (those endpoints supported by PingFederate 2). When replacing an existing PingFederate 2 server deployment, manually move this WAR to the `pingfederate/server/default/deploy/directory`.
- The PingFederate server can be configured to support the use of a proxy server when retrieving a CRL from a Certificate Authority. Relevant configuration settings reside in `pingfederate/server/default/data/config-store/revocation-checking-config.xml`.
- When the PingFederate server relies upon an external LDAP directory to authenticate administrative users, the `ldap.password` property in the `pingfederate/bin/ldap.properties` file now supports encrypted credentials (*Getting Started: Installation*).
- The PingFederate server allows imported SSL server certificates containing signatures from one or more intermediate Certificate Authorities. When SSL clients request an SSL connection to the PingFederate server, the entire SSL server certificate chain is presented.

- The Summary and Activation screens for both IdP and SP connections display a valid URL that serves as an example of a startSSO.ping endpoint used by local applications integrating with PingFederate (*Administrator's Manual: Managing SP Connections and Managing IdP Connections*).
- The Web SSO entry screens for both IdP and SP connections include summary information in a table describing relevant configuration settings (*Administrator's Manual: Managing SP Connections and Managing IdP Connections*).
- The PingFederate server prevents auditors from accessing links on the Main Menu that impact external resources. This includes exporting SAML metadata, signing XML files, and creating configuration archives (*Administrator's Manual: System Administration*).

PingFederate 5.0.2 – March 2008

- IdP Persistent Reference Cookie (IPRC) — Provides a mechanism allowing an SP PingFederate server to discover a user's IdP based on a persistent browser cookie that contains a reference to the IdP partner previously used for SSO.

This feature provides an alternative to standard IdP Discovery for SP-initiated SSO, as defined in the SAML specifications, which uses a common-domain cookie (CDC) written by the IdP (see the PingFederate *Administrator's Manual*). Unlike the IdP Discovery cookie, the IPRC is written by the SP PingFederate each time an SSO event for the user occurs (either IdP- or SP-initiated). The cookie identifies the IdP partner using information in the SAML assertion. For subsequent SP-initiated SSO requests, the SP server can skip a previously required step prompting the user to select an IdP for authentication when multiple IdP partners are configured but none is specifically identified in the SSO call received by the SP PingFederate server.

- Updated the IdP-selection template to make it easier to use. The new selection template is used when no IPRC (or CDC) is available and when there are multiple IdPs to which the user might have previously authenticated.
- Corrected an issue in which a Concurrent Modification Exception was encountered when server clustering is used and debug is turned on for log files. (The workaround for this issue in previous releases is to turn debug off.)
- When no certificate revocation list (CRL) is found during certificate validation checking, the subject certificate is assumed to be valid for the current SSO/SLO transaction. Previously, when no CRL was found, the certificate was deemed invalid and the transaction aborted. The default setting is changed for this release to prevent problems with upgrading to PingFederate 5.x from previous versions.

PingFederate 5.0.1 – January 2008

- Support for rapid provisioning of partner connections is available using Auto-Connect technology. Leveraging the existing SAML 2.0 specification, Auto-Connect allows PingFederate deployments to scale easily with minimal manual involvement. The majority of partner connection configuration occurs at runtime through the exchange of dynamically generated metadata (*Administrator's Manual: Managing SP Connections and Managing IdP Connections*).
- Administrators can authenticate to the administrative console using credentials in an external LDAP directory. This allows organizations with existing admin accounts to provide access to the

console without creating and managing individual accounts within PingFederate (*Getting Started: Installation*).

- Partner connections may be created by importing them programmatically into PingFederate through a SOAP interface. This allows administrators to provision partner connections without accessing the administrative console manually. The Connection Management screens (both IdP and SP) contain an “Export” action that creates an XML file containing a connection’s configuration (*Administrator’s Manual: Web Service Interfaces*).
- Server configuration data may be replicated to a cluster through a SOAP call to the administrative console. This allows cluster deployments to receive configuration changes without accessing the administrative console manually (*Administrator’s Manual: Web Service Interfaces*).
- The SAML 2 Attribute Query Profile is supported by PingFederate. This allows SPs to request user attributes from an IdP independent of user authentication (*Administrator’s Manual: Managing SP Connections and Managing IdP Connections*).
- Multi-valued attributes passed in an Assertion to a WS-Federation partner conform to how ADFS expects them.
- SAML metadata may be generated with a digital signature to guarantee authenticity (*Administrator’s Manual: System Administration*).
- The Protocol Endpoints popup contains online help links. These links may be used to learn more about the server’s endpoints from a partner perspective.
- The User-Session Creation screen in the IdP connection flow contains summary information that provides administrators with insight into the current configuration. Similarly, the Assertion Creation screen in the SP connection flow also provides administrators with useful configuration information (*Administrator’s Manual: Managing SP Connections and Managing IdP Connections*).
- Administrators can specify a descriptive “Connection Name” for partner connections (*Administrator’s Manual: General Information and General Connection Information*).
- The “Web SSO” portion of partner configuration is distinct from first/last-mile configuration (*Administrator’s Manual: Managing SP Connections and Managing IdP Connections*).
- Connection summary screens contain +/- buttons to show/hide major sections.
- Metadata import extracts the Base URL from the metadata file and populates relative URLs within the connection (*Administrator’s Manual: Importing Metadata and Importing IdP Metadata*).
- PingFederate communicates with an SNMP network-management console using standard SNMP Get and Trap operations (*Administrator’s Manual: Configuring Runtime Reporting*).
- The PingFederate engine exposes a URL designed for load balancers to determine whether a PingFederate server is available to process transactions (*Administrator’s Manual: Application Endpoints*).
- Certificate-management usability is improved (*Administrator’s Manual: Security Management*).
- Certificate expirations are tracked by PingFederate, and impending expirations may result in a notification sent via email, when configured (*Administrator’s Manual: Configuring Runtime Notifications*).
- New, more user-friendly sample applications focus on demonstrating PingFederate server functionality (*Quick-Start Guide*).

- The entry screen into the “Credentials” area of connections contains useful information about the credentials used (*Administrator’s Manual: Identity Provider SSO Configuration and Service Provider SSO Configuration*).
- The server ID is no longer displayed to the administrator.
- A cluster’s administrative console no longer aggregates transactions counts.
- The “Cluster Management” link replaces “High Availability” on the Main Menu (*Server Clustering Guide*).
- Clustering supports TCP and UDP, node authentication, optional encryption, and use of a single port for all communication (*Server Clustering Guide*).
- CRL processing updated to support the U.S. GSA’s E-Authentication v2 specification.
- The “About” pop-up contains additional license-key information.

PingFederate 4.4.2 – October 2007

Mitigated a number of potential security vulnerabilities regarding XML document processing

PingFederate 4.4.1 – June 2007

Addresses user-interface defects related to attribute query, XML encryption, and LDAP data store lookups

PingFederate 4.4 – May 2007

- Removal of support for the U.S. GSA’s E-Authentication v1.0 specification
- Addition for support of signed metadata files
- Support for partner certificate revocation through CRLs
- Increased flexibility around encryption of Name ID in SAML v2.0 SLO requests when the Name ID is encrypted within an assertion
- Support for the SOAP binding for both inbound and outbound SAML v2.0 messages
- Improved support for deployments where the server contains multiple network interfaces
- Usability enhancements to the Main Menu layout and Local Settings flow
- More sophisticated attribute-fulfillment operations through support of a Java-like syntax for data manipulation
- Removal of extraneous credentials settings for WS-Federation and SAML v1.x connections
- Improved display of long connection IDs on the Main Menu
- Inclusion of a demo application that complements the existing sample applications as described in the Quick-Start Guide

PingFederate 4.3 – March 2007

- Virtual Server Identities allow PingFederate to use distinct protocol identifiers in the context of a particular partner connection
- Additional customizable end-user error pages for 'page expired' and general unexpected error conditions
- Increased flexibility by allowing for a list of additional valid hostnames to be used for incoming protocol message validation
- Optionally, the SSO Directory Web Services can be protected with HTTP basic authentication
- New administrative console error page
- Improved short-term state management memory utilization for improved system resiliency
- Improved input-data validation and character-entity encoding of data when displayed--for protection against cross-site scripting attacks
- An IdP connection configured to use only a single SP Adapter Instance will ignore the URL-to-Adapter mapping step at runtime and just use the given adapter
- Blocked directory indexing to limit browsing of static web content
- Disabled unnecessary JRMP JMX port usage
- Mitigated HTTP response splitting attacks by disallowing potentially dangerous characters in all redirects

PingFederate 4.2 – December 2006

- Enhanced transaction logging functionality
- Sensitive user attribute values can be masked in log files to enhance privacy considerations
- The administrative console runs on a distinct port from the runtime engine allowing for more flexible and secure deployment options
- New filtering functionality on connection management screens enables easier management of large numbers of federation partners
- Adapter SDK enhancements to facilitate file downloads
- Usability refinements on X.509 certificate summary screens
- Less verbose description of certificates in drop down boxes improve look and feel
- Multiple partner endpoints of the same type can be configured to use the same binding
- Improved support for reverse proxy deployments

PingFederate 4.1 – October 2006

- Liberty Alliance interoperability certified
- SAML2 x509 Attribute Sharing Profile (XASP)

- Optional Hardware Security Module (HSM) mode, that enables storage of private keys and crypto processing on an external HSM unit that is FIPS-140-2 certified
- Updated Protocol Configuration Wizard
- Error handling templates that can be used to build SSO/SLO landing pages that communicate error status and support instructions to users
- Configuration options that enable multiple, simultaneous authentication profiles for the SOAP back-channel, including HTTP Basic, SSL Client Certificates, and Digital Signatures
- Digital signature capability for client authentication when using SAML 1.x
- Pop-up server endpoint display that filters by role and configurations made
- Two digital signature verification certificates can be assigned to a connection, allowing the partner flexibility in selecting one certificate or the other. When one certificate expires, the other certificate is used without the need for close synchronization.
- A `run.properties` configuration that allows an admin to specify an alternate port with which to communicate over the back-channel to partner's SAML gateway
- Support for 32- and 64- bit machine architectures

PingFederate 4.0 – June 2006

- Deploy multiple adapters as an IdP to look up different session security contexts across security domains and applications
- Save a partially completed connection as a draft
- Copy a connection to rapidly set up other partners or test environments with similar configurations
- Attribute source SDK enables retrieval of attributes from additional data source interfaces such as SOAP, flat files, or custom interfaces
- Multi-administrator support
- Ability to edit SP adapters that are in-use with target systems
- Encrypt or decrypt entire assertions or select elements, of particular value when intermediaries may handle SAML traffic
- Generate unique, Transient Name Ids each time the user federates to protect their identity
- SAML 2-compliant IdP Discovery mechanism that enables an SP to dynamically determine the appropriate IdP for the user
- Integration Kits provide additional methods that streamline passing of authentication context from an IdP to an SP
- Single log-out across all connections and protocols that support SLO
- Using an affiliate id, an SP can instruct an IdP to re-use the same persistent name identifier that was already used at other applications within the portal
- Non-normative support for SP-initiated SSO with SAML 1.x protocols

PingFederate 3.0.2 - February 2006

Upgrade of Jetty component to v5.1.10 in response to a security warning from the National Vulnerability Database

PingFederate 3.0.1 - December 2005

- Complete clustering support
- Optional email notification on licensing issues
- You can edit a previously configured connection (either IdP or SP) that uses a data store that is unavailable

PingFederate 3.0 – November 2005

- Support for SAML 2.0
- Use-case wizard for partner connection configurations
- Support for multiple security domains
- Redesigned user interface
- Embedded clustering
- Fixes for LDAP and JDBC connectivity

PingFederate 2.1 – July 2005

- Patched a concurrency bug in the XML security library
- Patched a memory leak in the XML-to-object binding library
- Removed the core protocol processor's reliance on a workflow engine to resolve a memory leak and improve overall performance
- Fixed a subtle memory leak in the module that tracks assertions in order to prevent replay in the POST profile
- Updated the default server SSL certificate (extended the expiration date)

PingFederate 2.0 – February 2005

Initial release