PingFederate® 6

Getting Started

PingIdentity®

Part Number 3007-398
Version 6.0
April, 2009

Ping Identity Corporation
1099 18th Street, Suite 2950
Denver, CO 80202
U.S.A.

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

Trademarks

Disclaimer

Contents

Preface **1**
About This Manual 1
Overview 1
Intended Audience 1
Text Conventions 2
Other Documentation 2
Chapter 1 Introduction **5**
About Identity Federation and SSO 5
Service Providers and Identity Providers 6
About WS-Trust STS 7
Enterprise Deployment Architecture 7
Use Case Configuration 8
Additional Features 8
Integration Kits 9
Token Translators 9
SaaS Connectors 10
Services and Support 10
PingEnable Methodologies 10
PingEnable Services 10
PingEnable Support 10
Chapter 2 Installation **11**
System Requirements 12
Operating Systems 12
User Store/Data Store Integration 12
Browsers 12
Java Environment 13
Minimum Hardware Requirements 13
Installing the JDK 13

Installing PingFederate 14
Running PingFederate for the First Time 15
Using LDAP Authentication 16
Deployment Options 17
Installing PingFederate as a Service 19
Uninstalling PingFederate 21
Uninstalling Services 22
Chapter 3 Console Navigation 23
Using the Main Menu 23
Navigating the Administrative Console 25
About Tasks and Steps 25
Console Buttons 26
Chapter 4 Supported Standards 29
Federation Roles 29
Terminology 30
SAML 1.x Profiles 32
SSO: Browser/POST 32
SSO: Browser/Artifact 34
SP-Initiated ("Destination-First") SSO 35
SAML 2.0 Profiles 36
Single Sign-on 36
SP-Initiated SSO: POST/POST 36
SP-Initiated SSO: Redirect/POST 38
SP-Initiated SSO: Artifact/POST 39
SP-Initiated SSO: POST/Artifact 41
SP-Initiated SSO: Redirect/Artifact 42
SP-Initiated SSO: Artifact/Artifact 43
IdP-Initiated SSO: POST 46
IdP-Initiated SSO: Artifact 47
Single Logout 48
About Session Clean-up 48
Attribute Query and XASP 48
IdP Discovery 49
WS-Federation 50
Passive Requestor Profile 50
Account Linking 51
Web Services Standards 52
Web Services Security 53
WS-Trust 54
Request Types 54
Transport and Message Security 56
Appendix A Using the SafeNet Luna HSM 57

Preface

About This Manual

This guide provides information about getting started with Ping Identity's PingFederate to deploy a secure Internet single sign-on (SSO) solution 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 SafeNet Luna HSM" — How to install and configure PingFederate with the Luna SA 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.
Italic	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

Unless otherwise noted, the documents listed below are located in your PingFederate installation's `pingfederate/docs` directory.

Tip: PingFederate provides context-sensitive online 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 example Web applications. Ping Identity recommends that you follow this *Guide* as a first step to establishing a simple identity federation between two Web applications and to familiarize yourself with PingFederate. The *Guide* is located in the `quickstart/docs` directory.

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 version 4 and higher. This G*uide* is located in the `pingfederate/sdk` directory.

Web Resources – Ping Identity continually updates its Web site with general and technical information in the form of White Papers, FAQs, Tech Notes, and other resources—www.pingidentity.com.

PingFederate documents may include hypertext links to 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 (support@pingidentity.com) if you encounter a problem.

Chapter 1

Introduction

Welcome to PingFederate, Ping Identity's rapidly deployable solution for secure Internet single sign-on (SSO), Web-Services identity access, and cross-domain user management.

Using standards-based identity federation, PingFederate provides an organization's network users with secure access to Web applications or other Internet resources, including identity-enabled Web Services, without the need for repeated logons. PingFederate thus reduces or eliminates repetitious user provisioning and time-consuming proprietary SSO or Security Token Service (STS) implementations.

For organizations needing secure SSO to Software-as-a-Service (SaaS) providers, PingFederate also offers optional connection templates and automated user provisioning for selected SaaS applications (including Salesforce and Google Apps). This option allows you to leverage existing identity-management investments—thus eliminating the need to replicate and expose any confidential user data over the Internet (see "SaaS Connectors" on page 10).

About Identity Federation and SSO

Federated identity management (or "identity federation") enables enterprises to exchange identity information securely across Internet domains, providing secure SSO. Federation is also used to integrated 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 secure SSO is desirable.

Note: This manual and other PingFederate manuals often refer to secure Internet SSO as "browser-based SSO" or just "Browser SSO" (as displayed in the administrative console). These terms are used to differentiate Internet SSO, which relies on the user's browser to transport messaging via HTTP, from single sign-on used for Web Services access, which does not require HTTP.

This cross-domain, identity-management solution provides numerous benefits, ranging from enhanced customer relations and service 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 29.

Service Providers and Identity Providers

Identity federation standards identify two operational roles in an Internet 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 Internet 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 (see Figure 1).

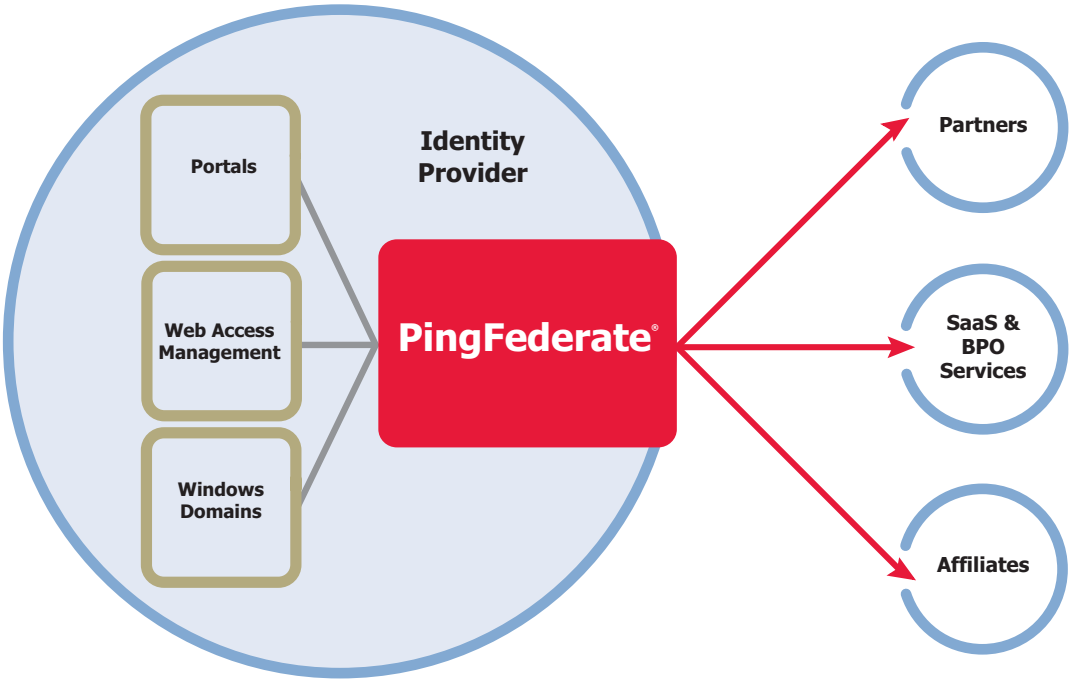

Figure 1: Secure Internet Single Sign-on

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.

About WS-Trust STS

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 52.)

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 (Security Assertion Markup Language) 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*.)

Enterprise Deployment Architecture

PingFederate is the only federated identity server that enables secure SSO and access to identity-enabled Web Services between applications residing in multiple security domains using different protocols. This effectively allows you to manage your partner trust relationships and connections from a single location.

PingFederate fully supports Burton Group's recommended architecture for a stand-alone server, thus enabling efficient, platform-level scalability for all your Internet SSO and Web-Service-access initiatives.

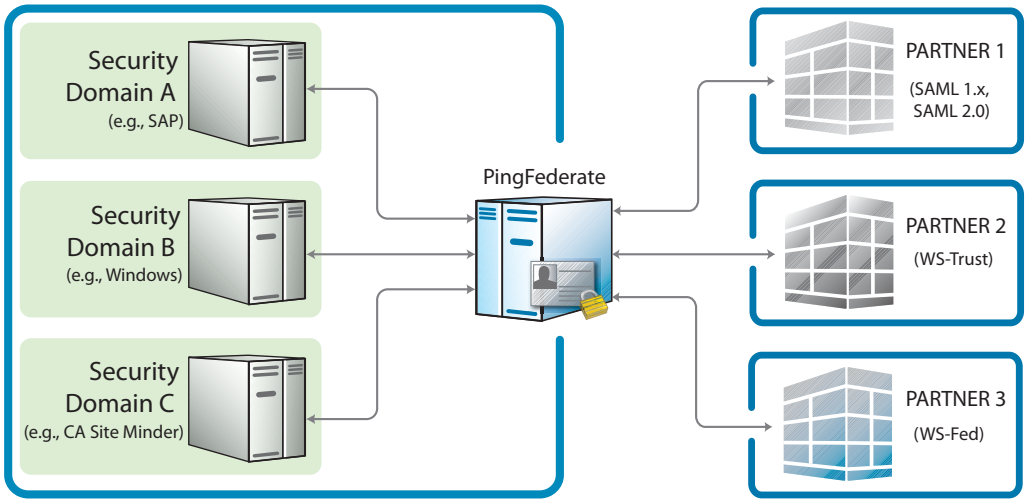

Figure 2: Multiple security-domain, multi-protocol federation

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's enterprise-deployment architecture 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.

Additional Features

PingFederate's lightweight, stand-alone architecture means you can receive the benefits of standards-based Internet SSO and Web Services identity-management 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.

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.

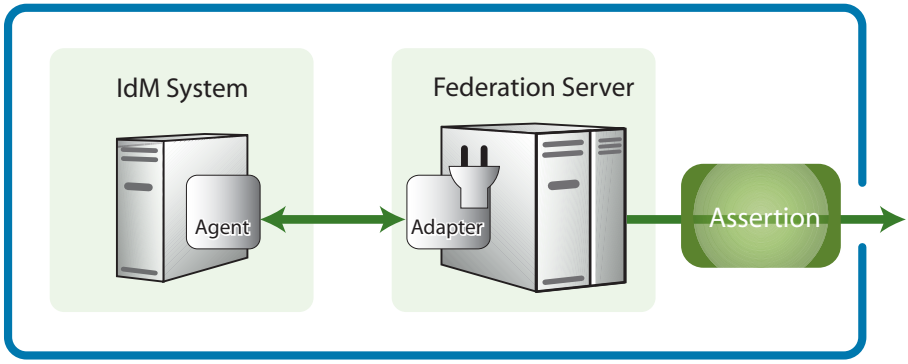

Figure 3: Integration Kit Architecture

Integration kits enable rapid session integration with both existing authentication services and target applications. (Figure 3 illustrates first-mile IdM system integration.) In addition, PingFederate includes a Software Development Kit for creating custom integrations.

Visit pingidentity.com for the latest information on availability of our integration kits.

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 information on all available token translators, visit pingidentiy.com.

SaaS Connectors

PingFederate's SaaS Connectors offer a streamlined approach for secure SSO to selected SaaS providers—including automatic user provisioning and deprovisioning (see "SaaS Provisioning" in the "Key Concepts" chapter of the PingFederate *Administrator's Manual*). The Connector packages include quick-connection templates, which automatically configure endpoints and other connection information for each provider.

Services and Support

Ping Identity's PingEnable® provides customizable services designed to meet the specific needs of customers' identity-management environments, using proven support experience and practices.

PingEnable Methodologies

Born from lessons learned and practices mastered during hundreds of successful deployments, PingEnable's detailed methodologies guide information technology (IT) organizations and line-of-business owners through each step of the identity-federation process. PingEnable methodologies are available to all customers and can be used individually or in conjunction with other PingEnable services. The methodologies include:

- Evaluation
- Quick-Start Implementation
- Managing Connection Partners
- Custom Application Interfacing

PingEnable Services

On-site and remote service offerings are available for each step of the identity-federation process, including capacity planning, system tuning, code reviews, and environmental optimization. Education and training services are also available, either remotely or on site.

PingEnable Support

Ping Identity is dedicated to helping customers successfully deliver secure Internet SSO. We have developed three custom levels of support to best meet different needs.

If you have an existing support contract with Ping Identity, please submit support requests at support.pingidentity.com, or call the phone number listed in your "Welcome to PingFederate Support" email.

If you do not have an existing support contract and you would like to discuss your service options, please contact sales@pingidentity.com or call toll-free 877.898.2905 (+1 303.468.2882 outside North America).

Chapter 2

Installation

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

This chapter covers:

- "System Requirements" on page 12
- "Installing the JDK" on page 13
- "Installing PingFederate" on page 14
- "Running PingFederate for the First Time" on page 15
- "Using LDAP Authentication" on page 16
- "Deployment Options" on page 17
- "Installing PingFederate as a Service" on page 19
- "Uninstalling PingFederate" on page 21

System Requirements

PingFederate is supported for deployment and configuration with the system specifications below.

Note: PingFederate functions normally under a variety of platform configurations, including Web browsers, not specified below. Also, data stores may potentially include any LDAP v3-compatible directory service or JDBC-compatible database.

Platform and data-store testing and qualification are ongoing; consult PingFederate Supported Platforms (www.pingidentity.com/products/supported-platforms.cfm) for the latest information.

Operating Systems

Windows

- Microsoft Windows Server 2003 with Service Pack 2 on x86 (32-bit)
- Windows XP Professional with Service Pack 2 (32-bit)

Linux

- Red Hat Enterprise Linux ES 4.2 with 2.6.9-22.0 Kernel on x86 (32-bit)

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.

User Store/Data Store Integration

JDBC Compatible

- Oracle 10g (on Windows 2003)

LDAP v3 Compatible

- Active Directory 2003 (with SP 1)
- Sun Directory Server 5.2

Browsers

- Internet Explorer 7.0
- Firefox 2.0

Browsers must be JavaScript-enabled.

Java Environment

The Java SE Development Kit (JDK) 1.6.

Important: The JDK must be installed in a path containing no spaces. For example, do *not* use the "Program Files" folder on Windows.

Minimum Hardware Requirements

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

Installing the JDK

The JDK 1.6 provides the required environment for PingFederate.

Note: If your site requires compliance with Federal Information Processing Standard (FIPS) 140-2, then the JDK 5 is required (see "Using the SafeNet Luna HSM" on page 57).

Important: You must install the JDK before installing PingFederate.

To install the JDK for Windows and Linux:

1. Download the JDK at: java.sun.com/javase/downloads/index.jsp.
2. Install the JDK to a location with no spaces in the path (for example, `C:\j2sdk1.6`).
3. 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 104-2, see "Using the SafeNet Luna HSM" on page 57 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 into your system with appropriate privileges to install and run an application.

2. Verify that the JDK is installed and environment and PATH variables are set correctly (see "Installing the JDK" on page 13).

3. Create an installation directory.

   Important: The installation path and the directory name must *not* contain spaces.

4. Extract the distribution ZIP file into the installation directory.

5 Request a license key.

   Sign on to the Ping Identity Web licensing page (www.pingidentity.com).

6 Save the license key file in the directory:

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

   Ensure the file is named:

   `pingfederate.lic`

   Note: If you are deploying PingFederate in a cluster configuration, you may install the license key on any server in the cluster. (For more information, see the PingFederate *Server Clustering Guide*.)

Running PingFederate for the First Time

The first time you run the PingFederate administrative console, you normally use a default username and password supplied with the distribution.

Tip: If your network uses LDAP to authenticate users, you can choose to set up PingFederate to use the network data store (see "Using LDAP Authentication" on page 16). After completing this setup, use a valid network username and password to log on to PingFederate for the first time (and subsequently), rather than the defaults.

After you launch the administrative console and log on, you must change the default password (unless you are using LDAP authentication). 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 23).

Note: You can change the installation setup via menu choices under My Server on the Main Menu (see the "System Settings" chapter in 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`

   (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:
   `Started in XXs:XXms`

2. 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*.

3. Enter Username and Password.

   If you have configured PingFederate to use your network authentication system, enter your network credentials. Otherwise, enter the installed, first-time defaults:

   Username: `Administrator`

   Password: `2Federate`

4. If prompted, 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*.

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

   For more information see sections under "Managing Server Settings" in the "System Settings" chapter of the PingFederate *Administrator's Manual*.

Using LDAP Authentication

You can configure PingFederate to use your network's LDAP user-data store for authentication to the administrative console, as an alternative to using PingFederate's own internal data store. The LDAP authentication setup is available via configuration files in the `<pf_install>/pingfederate/bin` directory.

You can configure this feature at any time. Note that user-management functions are handled outside the scope of the PingFederate administrative console when LDAP authentication is enabled.

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`.

2. In the `pingfederate/bin/ldap.properties` file, change property values as needed for your LDAP 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" in the "System Administration" 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.

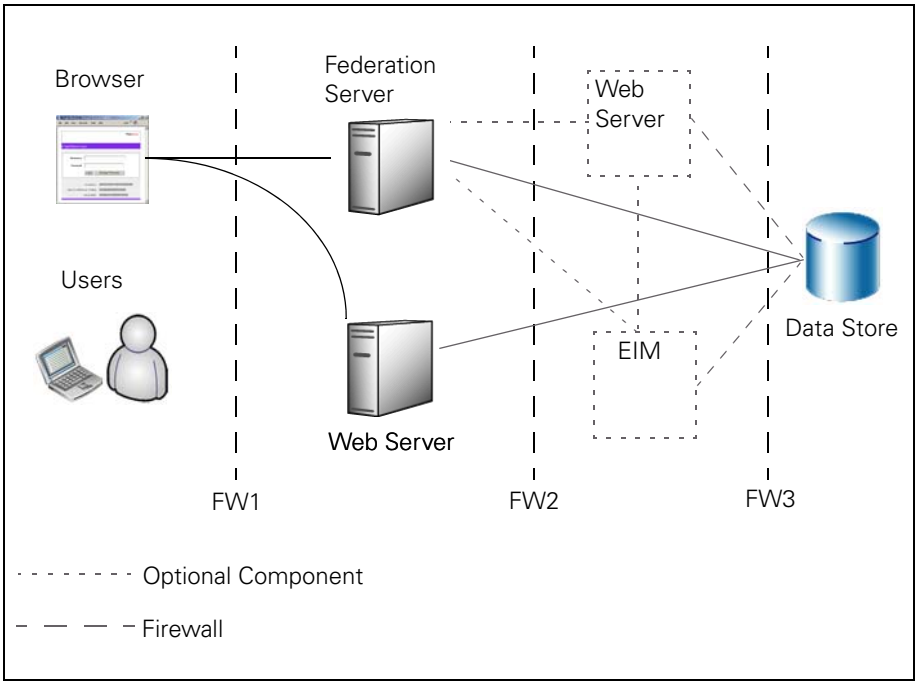

Figure 4: Stand-alone Deployment Example

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.

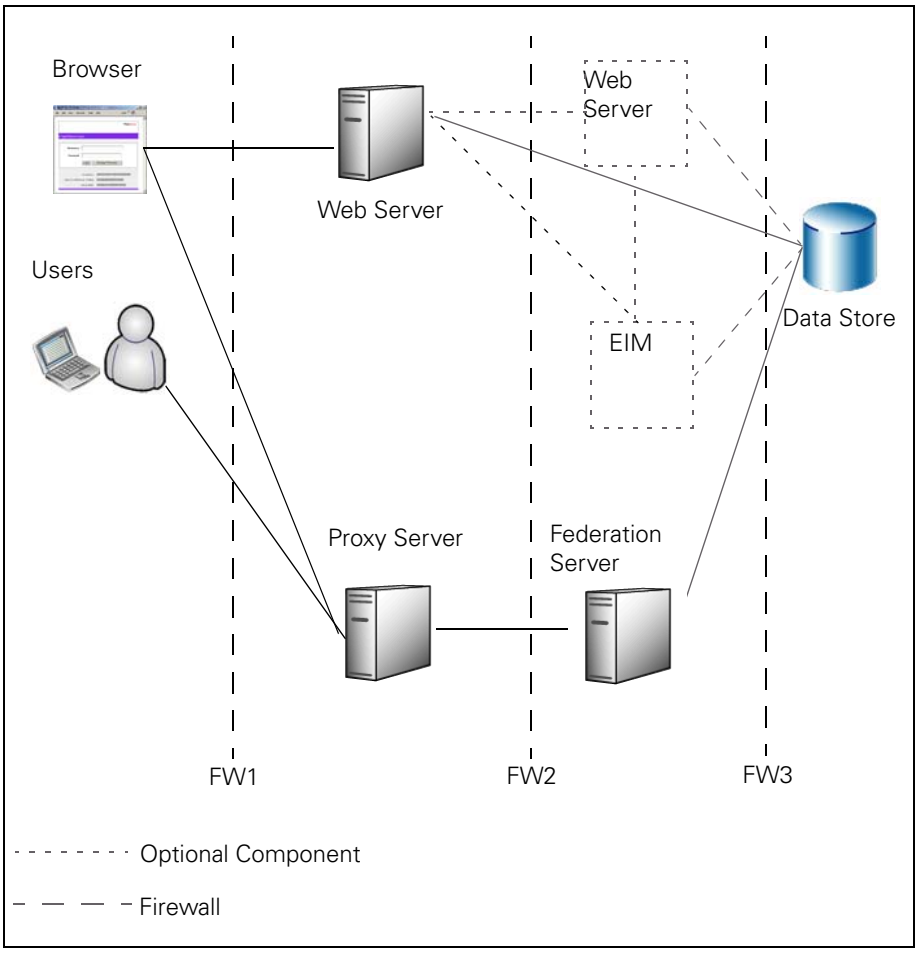

Figure 5: Proxy Deployment Example

Installing PingFederate as a Service

You can set up PingFederate to run in the background as a service on either Windows or Linux running 32- or 64-bit processors.

Note: Before performing this procedure, ensure that PingFederate runs normally by manually starting the server (see "Running PingFederate for the First Time" on page 15).

(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 22).

To install PingFederate as a service:

1. Complete the steps under "Installing PingFederate" on page 14.
2. Ensure you are logged on with full Administrator privileges.
3. To install the service on a 32-bit Windows platform, run `install-service.bat` from the directory:

   `<pf_install>\pingfederate\sbin\win-x86-32`

   Or:

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

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

   Or:

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

   `<pf_install>\pingfederate\sbin\win-itanium-64`
4. Access the Windows **Control Panel** > **Administrative Tools** > **Services**.
5. 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 install PingFederate as a service on Linux, you must place a script in the system initialization directory. Before running PingFederate as a service,

manually start the PingFederate server to ensure that it is configured properly (see "Running PingFederate for the First Time" on page 15).

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 14.
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.
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.

This script, modified to work with PingFederate, is based on the script *StartJBossOnBootWithLinux* (see http://wiki.jboss.org/wiki/StartJBossOnBootWithLinux).

```
#! /bin/sh

start(){
    echo "starting PingFederate.."
    su - <PF_user> -c '<pf_install>/sbin/pingfederate-run.sh
    > /dev/null 2> /dev/null'
}
stop(){
    echo "stopping PingFederate.."
    su - <PF_user> -c '<pf_install>/sbin/pingfederate-
    shutdown.sh -S'
}

restart(){
    stop
# padding time to stop before restart
    sleep 60
# protect against any services that are not stopped
# (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.

Note: If you are uninstalling the service for PingFederate 5.2 or a previous version, use the script `uninstall-pf-svc.bat` located in the `..\sbin\legacy` directory.

(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 "Installing PingFederate as a Service" on page 19).
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 "Installing PingFederate as a Service" on page 19).

Chapter 3

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 23
- "Navigating the Administrative Console" on page 25

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 24).

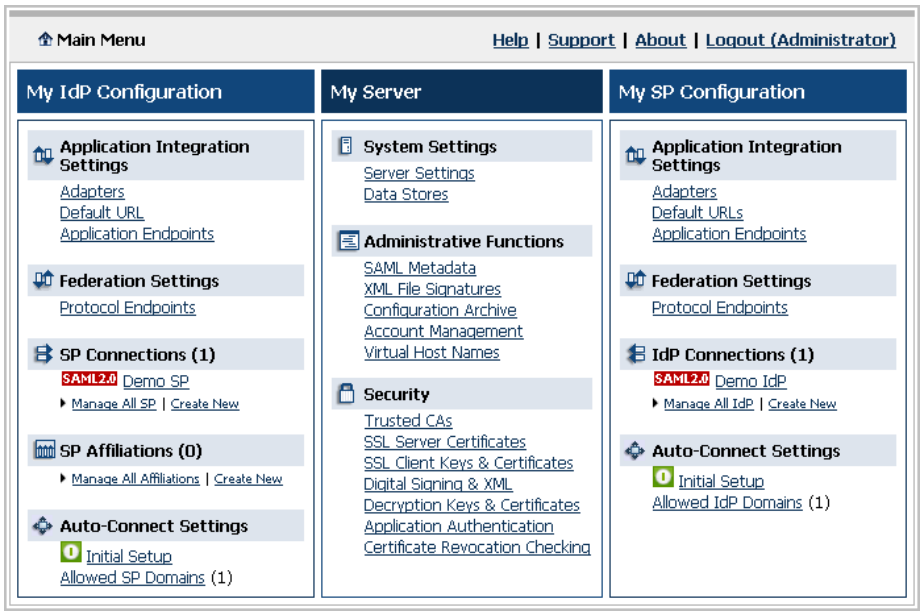

Figure 6: Main Menu (Example)

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*
- Manage certificates and application authentication for attribute query requests—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 26).

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.

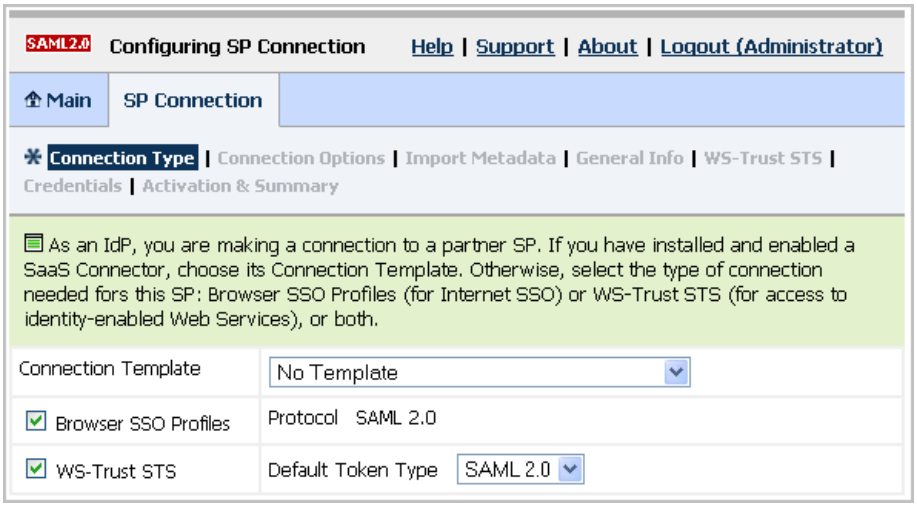

Figure 7: Tasks and Steps (Example)

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** 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 steps 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 CAs" 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 25). 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, continue the configuration until you see a **Save** button or 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** or **SP]** under [IdP or SP] Connections on the Main Menu and then select the draft from the connection list.

Table 2: Administrative Console Buttons (Continued)

Button	Description
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 25).
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.

Chapter 4

Supported Standards

PingFederate provides flexible, integrated support for all versions of the Security Assertion Markup Language (SAML) protocol, from 1.0 through 2.0, 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 29
- "Terminology" on page 30
- "SAML 1.x Profiles" on page 32
- "SAML 2.0 Profiles" on page 36
- "WS-Federation" on page 50
- "Account Linking" on page 51
- "Web Services Standards" on page 52
- "Transport and Message Security" on page 56

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 49).

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 (SAML 2.0)
- Authentication Context (SAML 2.0)

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 32.

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 Chapter 2 of the PingFederate *Administrator's Guide*.)

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 Chapter 2 of the PingFederate *Administrator's Guide*).

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 exact content of the authentication context is left up to the federation partners to interpret and implement. Each IdP will use a different set of authentication technologies, follow different processes, and be bound by different legal obligations regarding authentication.

The SAML specification provides an XML schema whereby partners can create authentication context declarations. Partners may choose to implement a set of classes provided by the specification to help categorize and simplify context interpretation. Partners may choose to reference a SAML 2.0 URN that identifies each of these classes (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.

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 Chapter 1 of the PingFederate *Administrator's Guide*). 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. The OpenToken Adapter packaged with PingFederate supports this feature.

For more information on configuring authentication context for an adapter instance, see "Selecting an Authentication Context" in Chapter 5 of the PingFederate *Administrator's Guide*.

SAML 1.x Profiles

SAML 1.0 and 1.1 profiles provide for secure Internet 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.

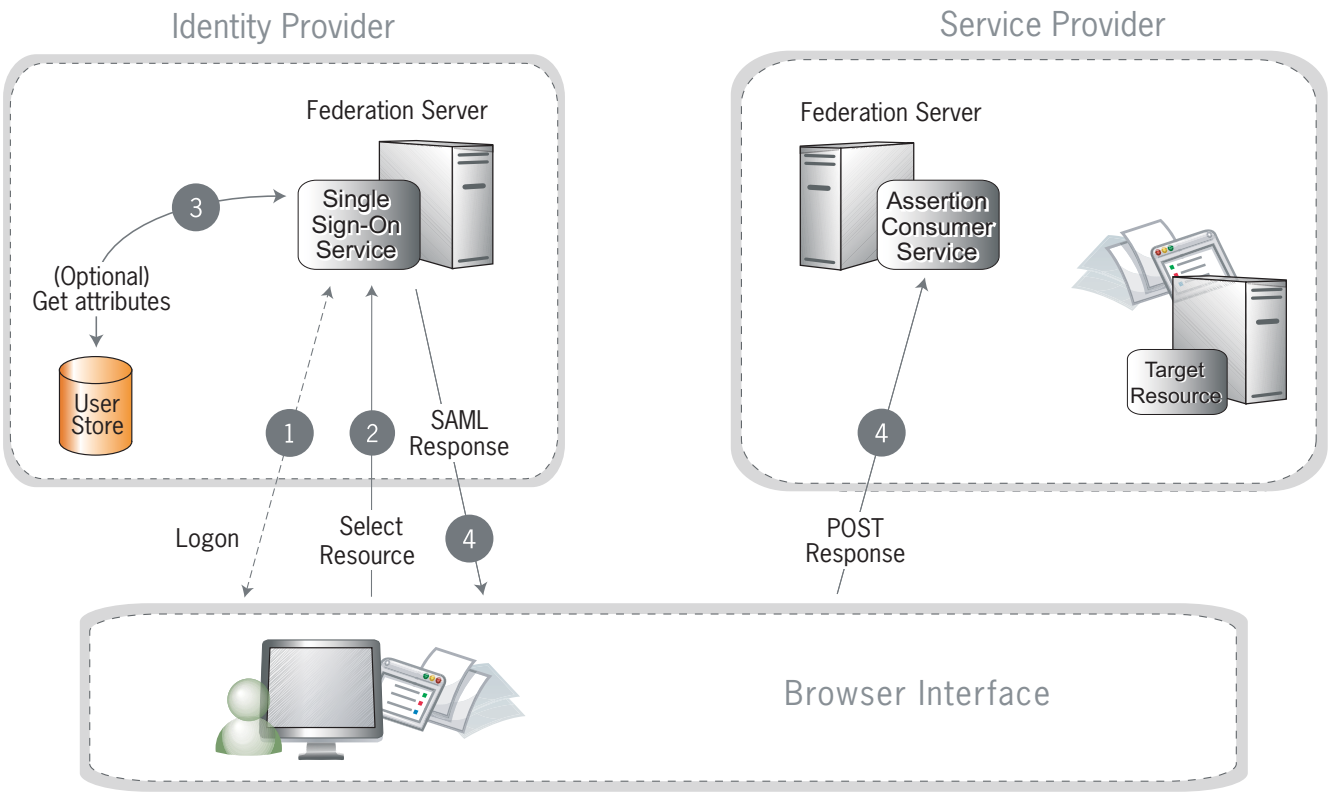

Figure 8: Browser/POST Profile

Processing Steps:

1. A user has logged on to the IdP.
2. The user requests access to a protected SP resource. The user is not logged on to the SP site.
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.

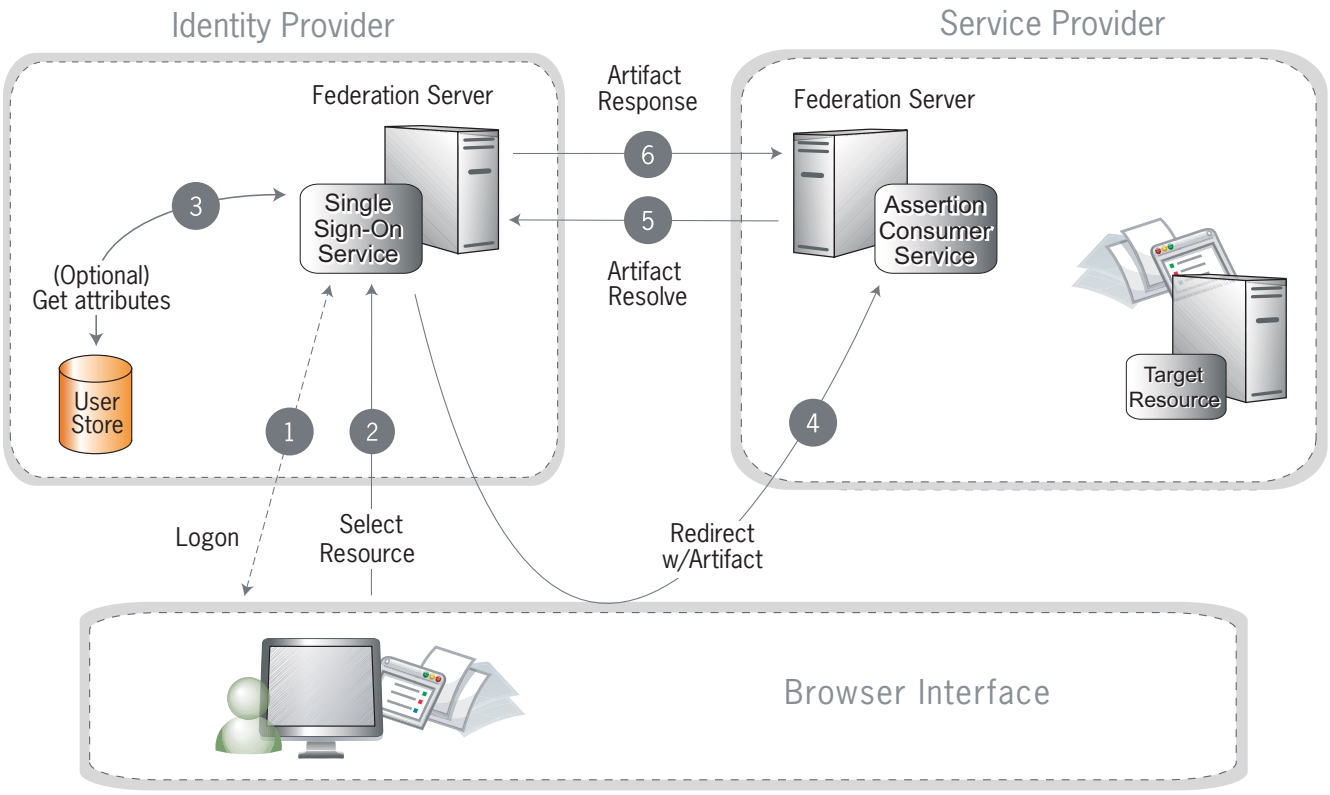

Figure 9: SSO: Browser/Artifact Profile

Processing Steps:

1. A user is logged on to the IdP.
2. The user requests access to a protected SP resource. The user is not logged on to the SP site.
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.

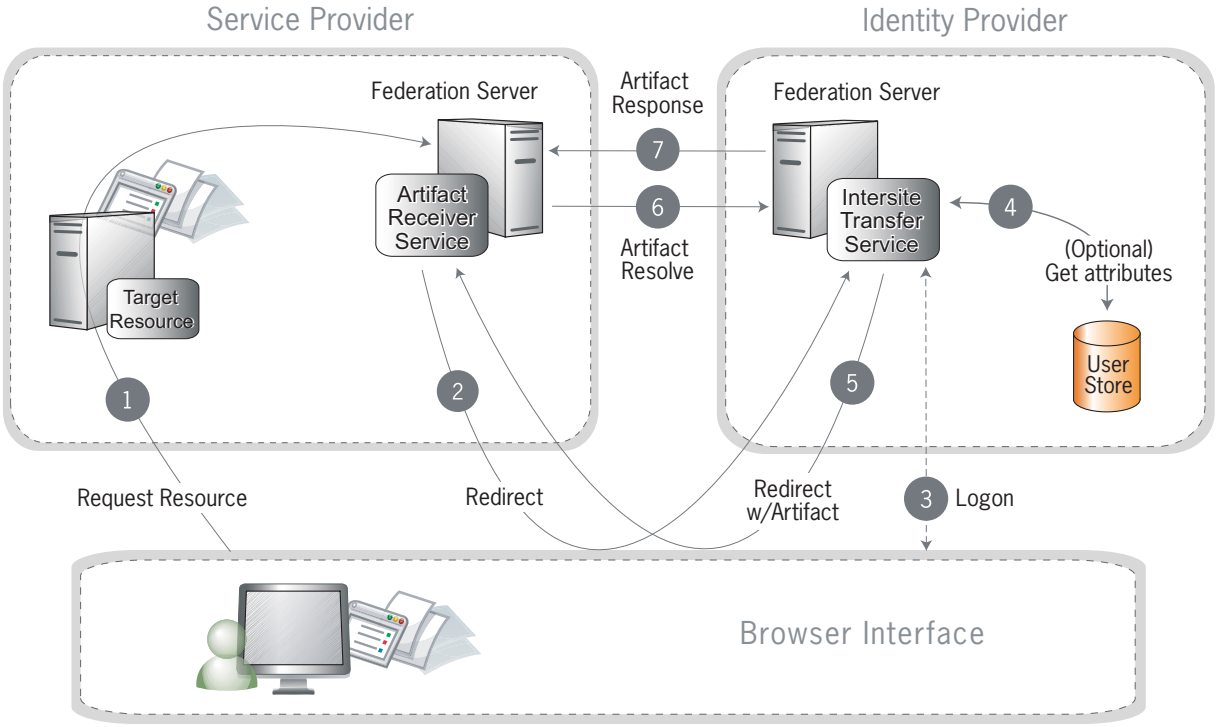

Figure 10: SP-Initiated SSO

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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.

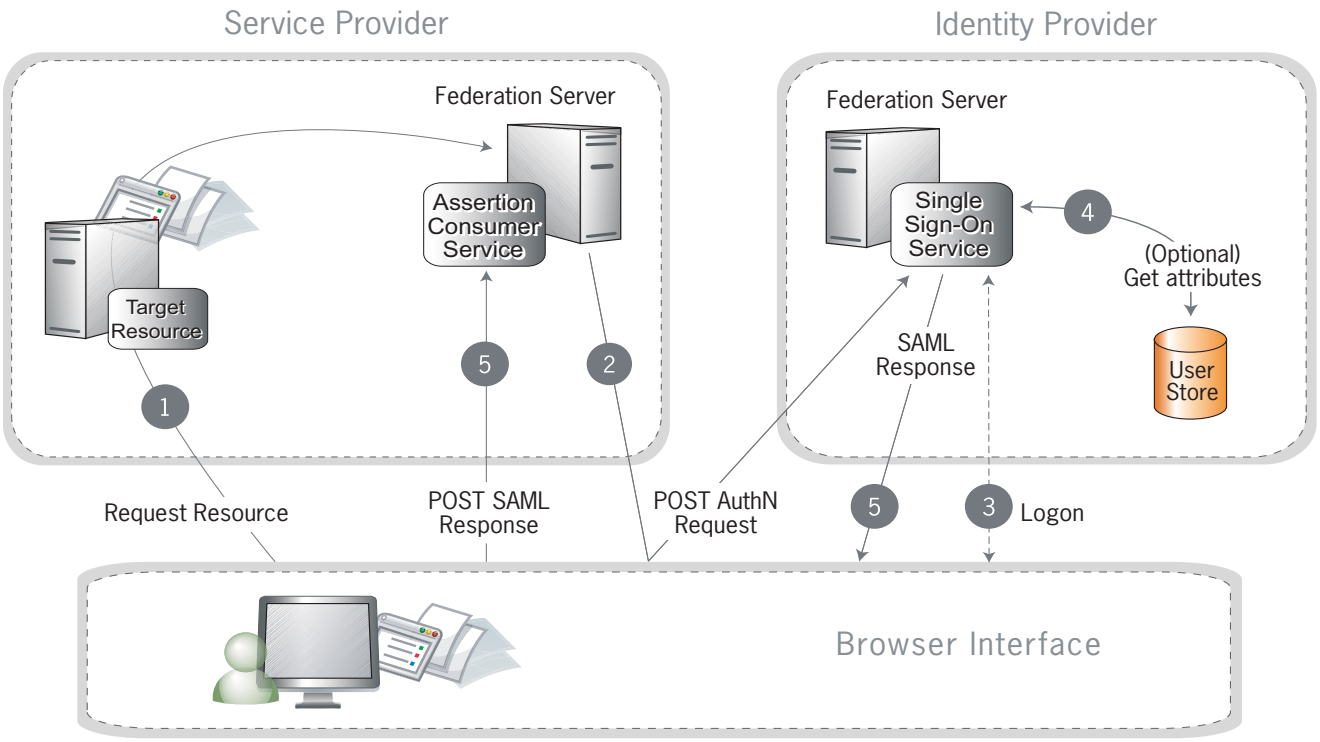

Figure 11: SP-Initiated SSO: POST/POST

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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: 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.

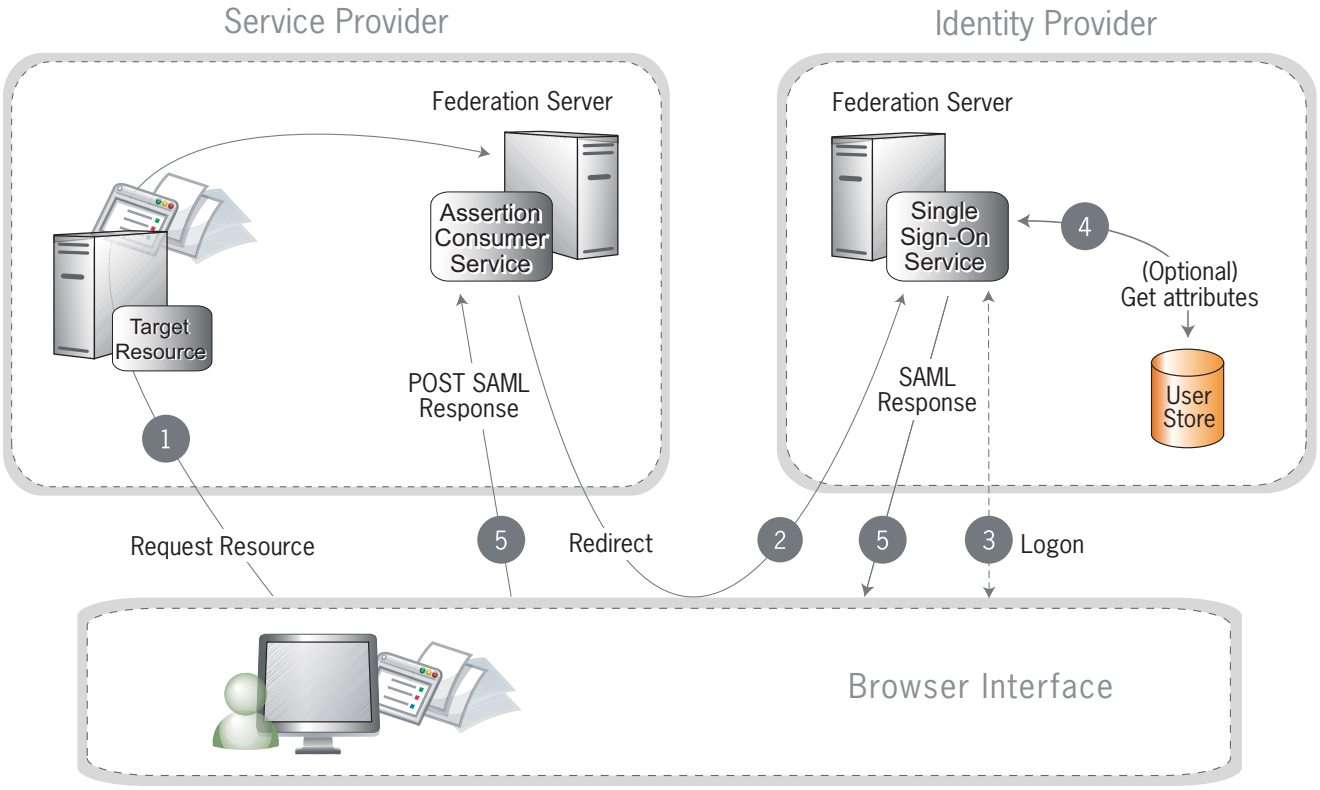

Figure 12: SP-Initiated SSO: Redirect/POST

Processing Steps:

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 Chapter 1 of the PingFederate *Administrator's Guide*.)

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.

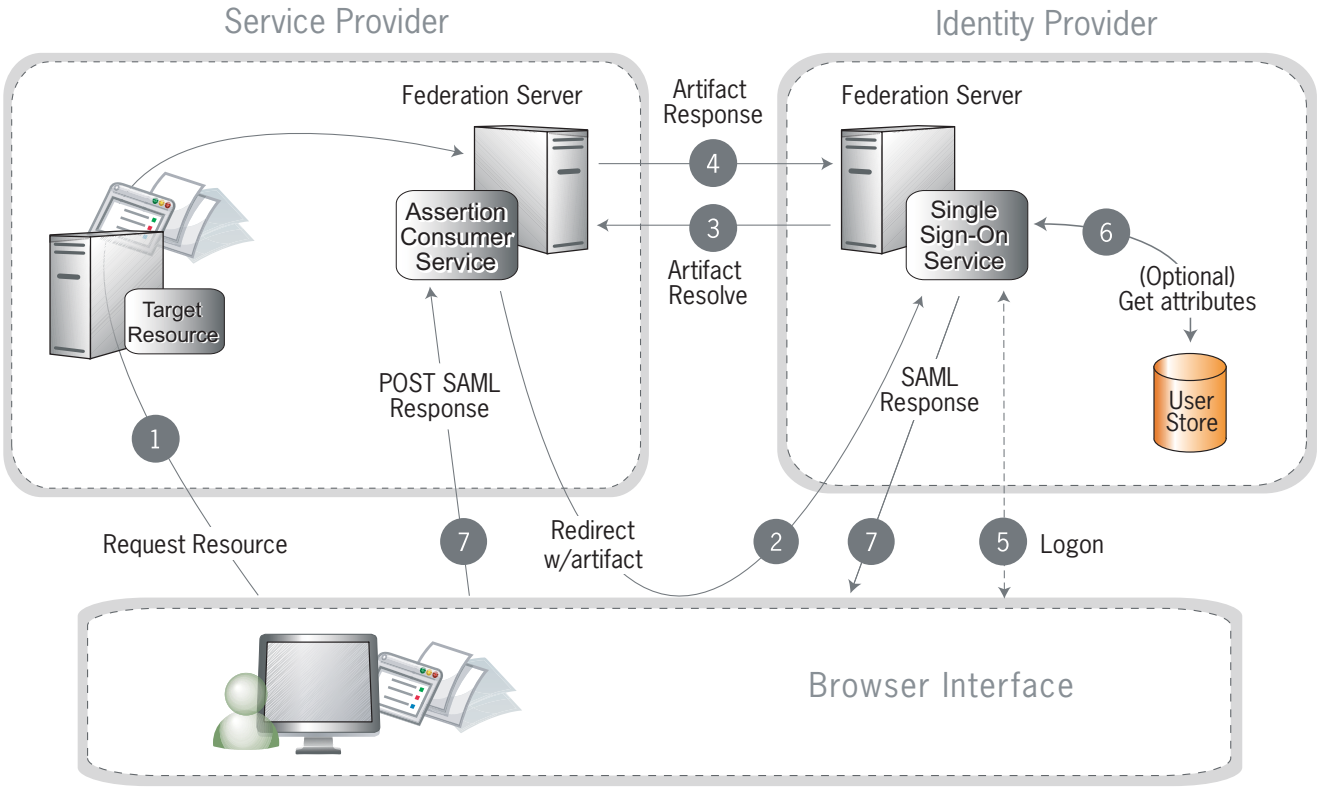

Figure 13: SP-Initiated SSO: Artifact/POST

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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.

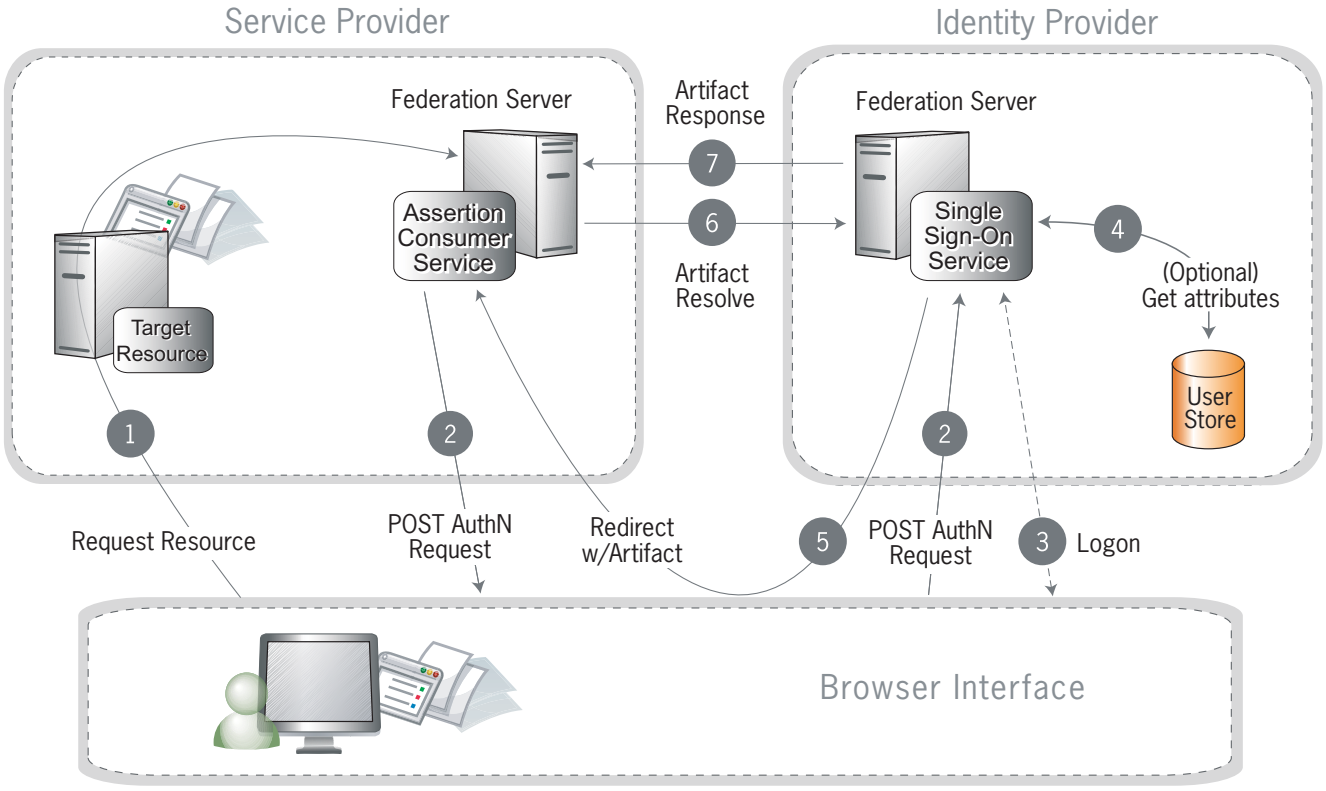

Figure 14: SP-Initiated SSO: POST/Artifact

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 Chapter 1 of the PingFederate *Administrator's Guide*.)

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.

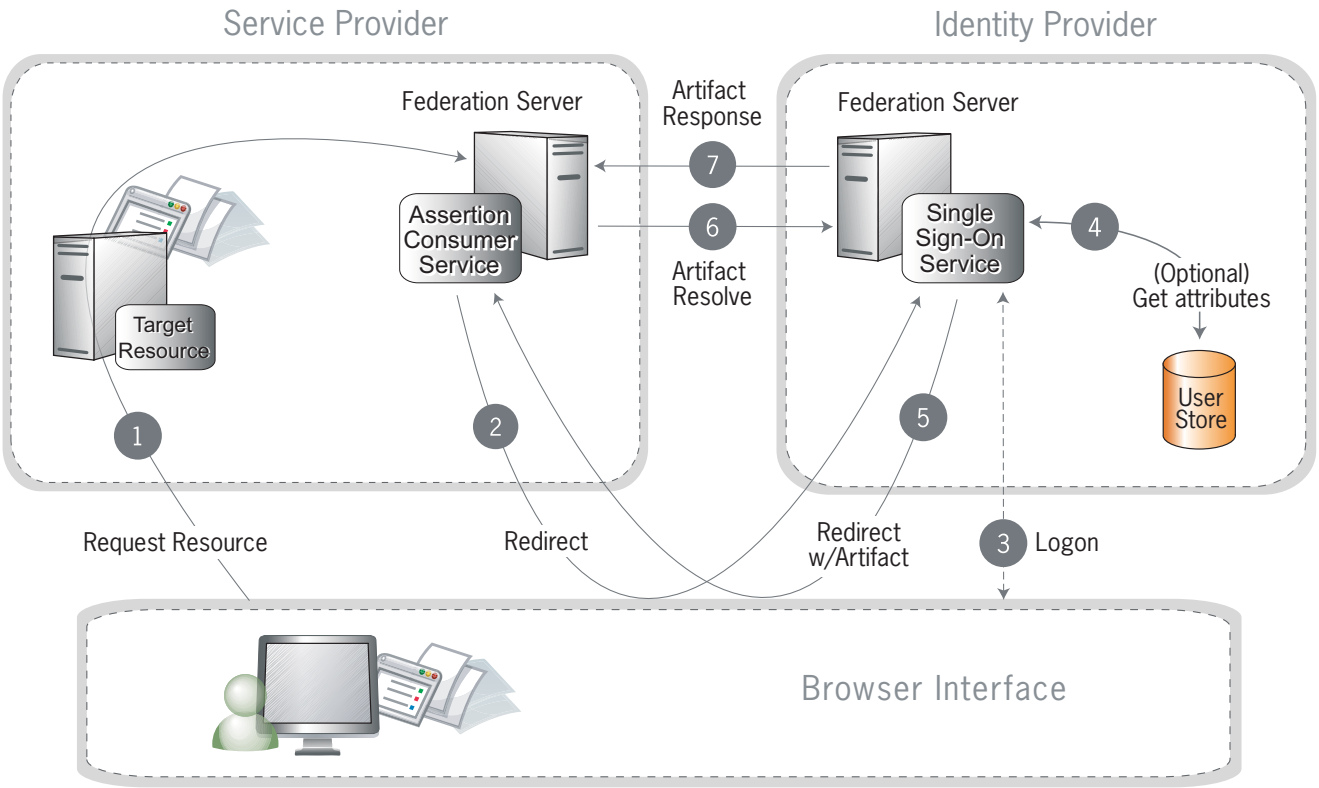

Figure 15: SP-Initiated SSO: Redirect/Artifact

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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.

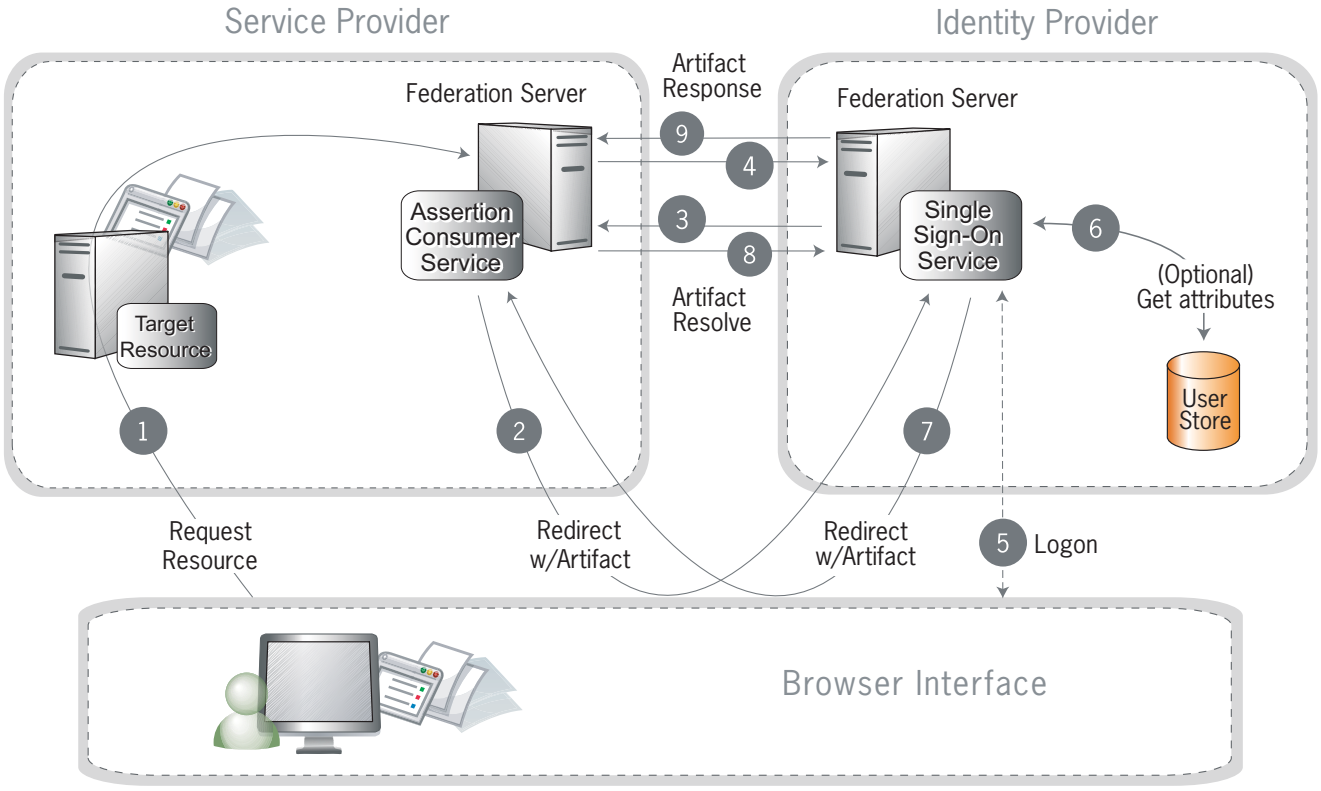

Figure 16: SP-Initiated SSO: Artifact/Artifact

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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.

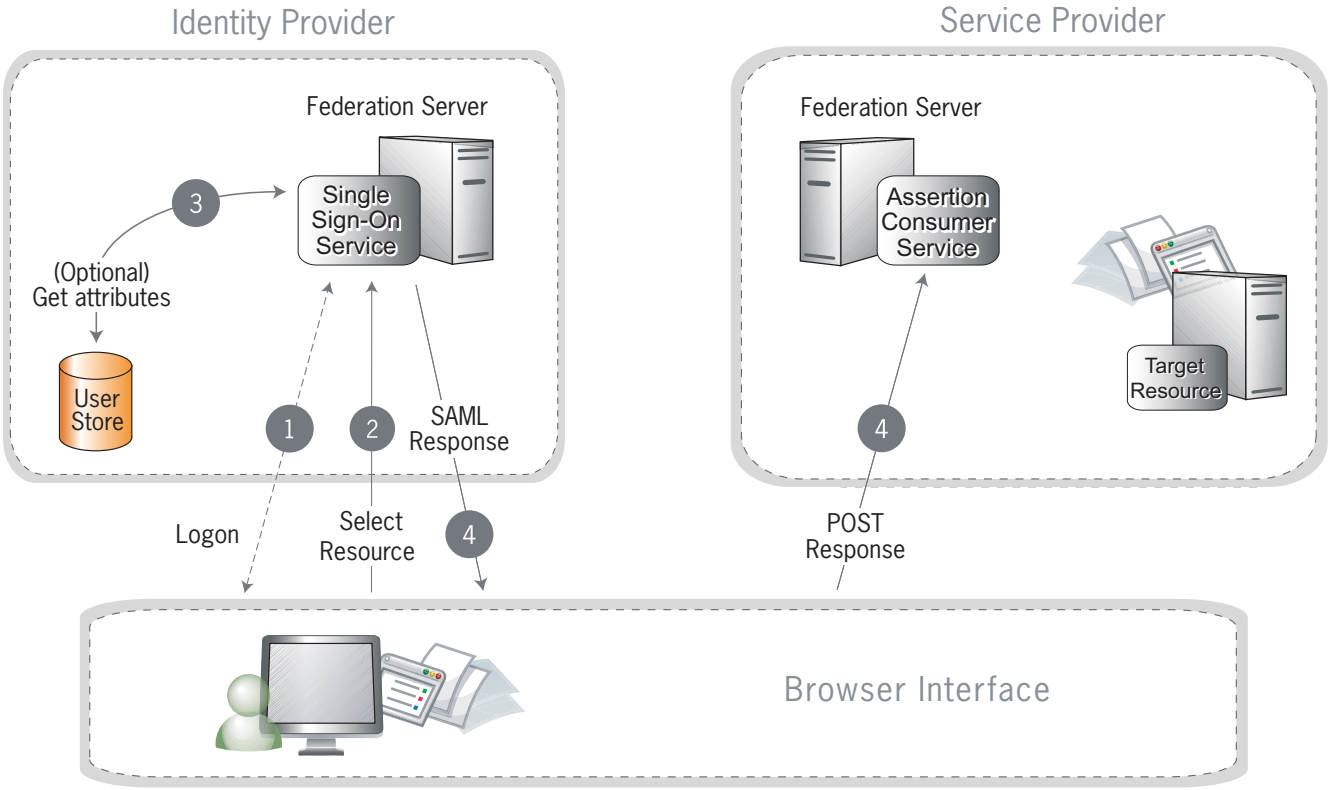

Figure 17: IdP-Initiated SSO: POST

Processing Steps:

1. A user has logged on to the IdP.
2. The user requests access to a protected SP resource. The user is not logged on to the SP site.
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.

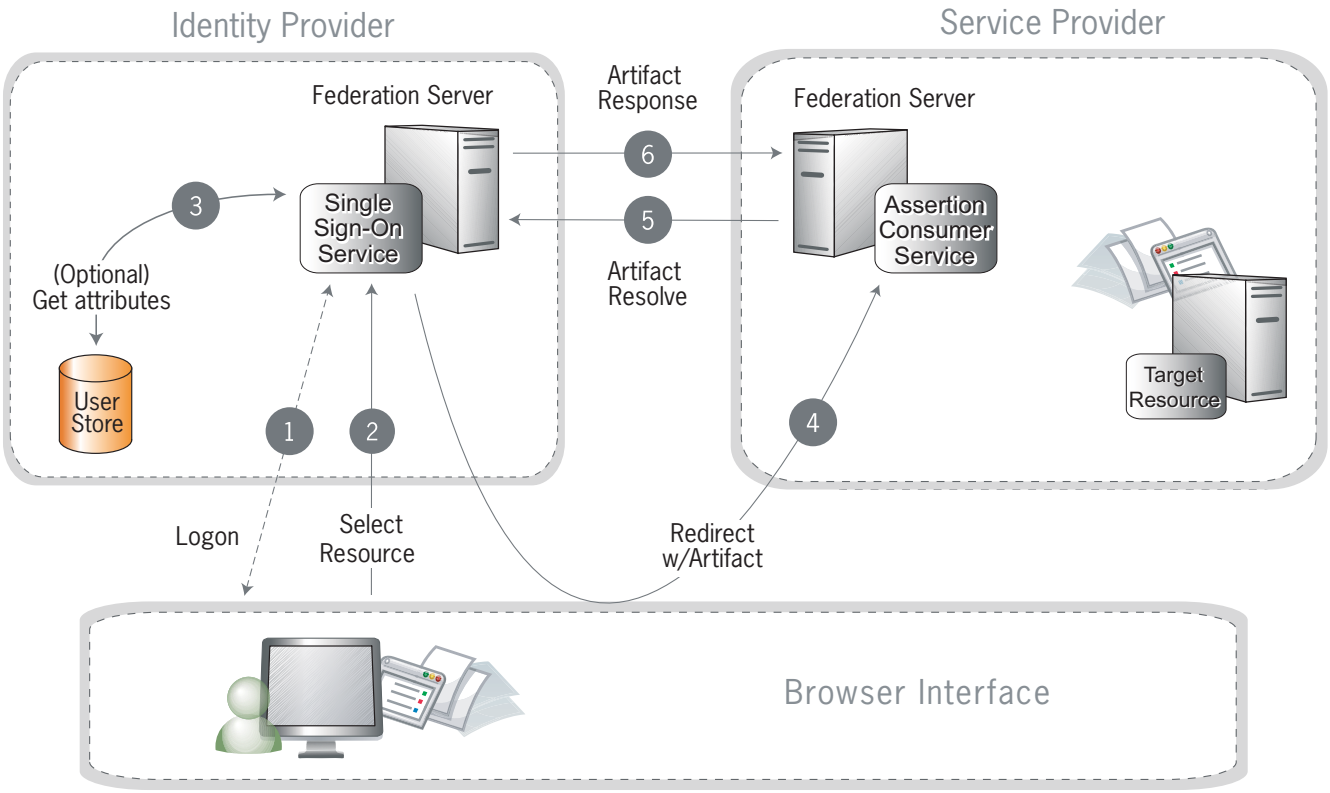

Figure 18: IdP-Initiated SSO: Artifact

Processing Steps:

1. A user is logged on to the IdP.
2. The user requests access to a protected SP resource. The user is not logged on to the SP site.
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 36 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 Chapter 1 of the PingFederate *Administrator's Guide*.

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 Chapter 1 of the PingFederate *Administrator's Guide*).

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 "Application Authentication" in Chapter 4 of the PingFederate *Administrator's Guide*). 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 Chapter 6 of the PingFederate *Administrator's Guide*). 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 Chapter 1 of the PingFederate *Administrator's Guide*).

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 Chapter 3 of the PingFederate *Administrator's Guide*.

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" at:
http://www-128.ibm.com/developerworks/library/specification/ws-fed/

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.

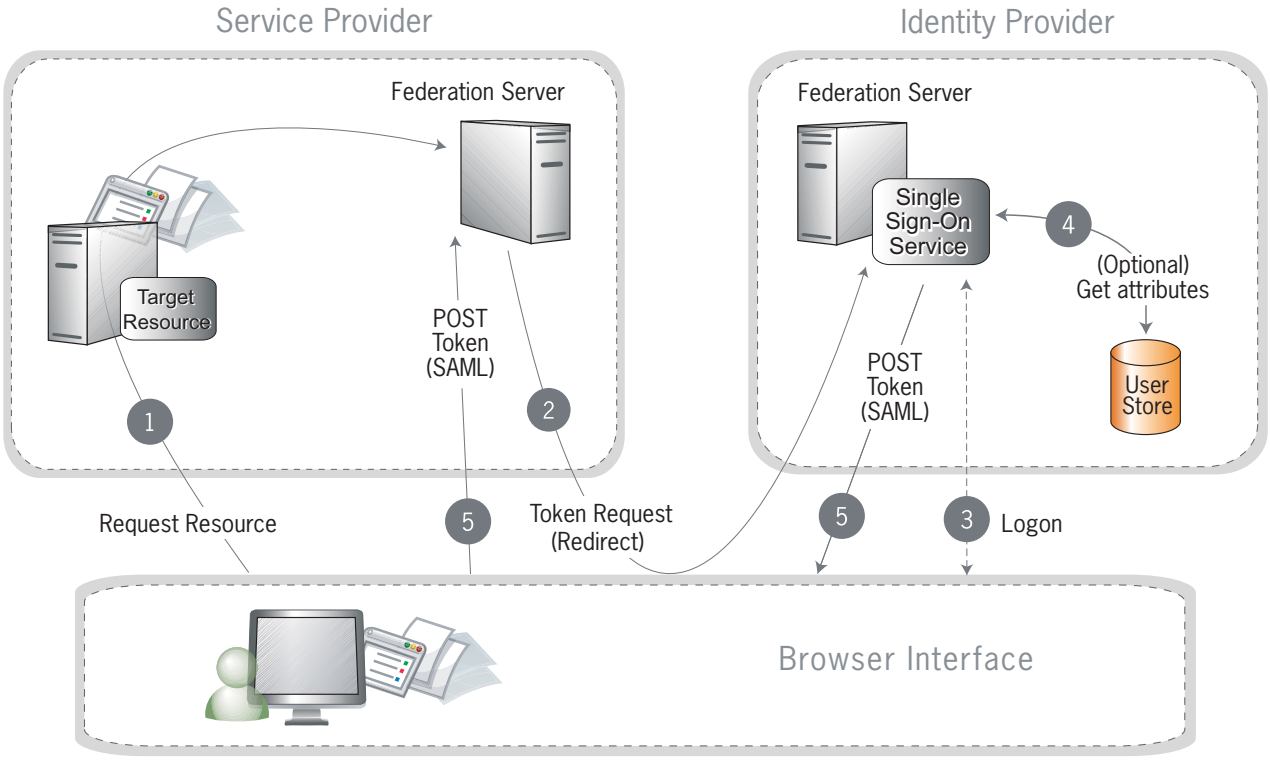

Figure 19: WS-Federation SSO

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 Chapter 1 of the PingFederate *Administrator's Guide*.)
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 48); however, HTTP GET/POST is always used as the transport mechanism.

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 Chapter 1 of the PingFederate *Administrator's Guide*.

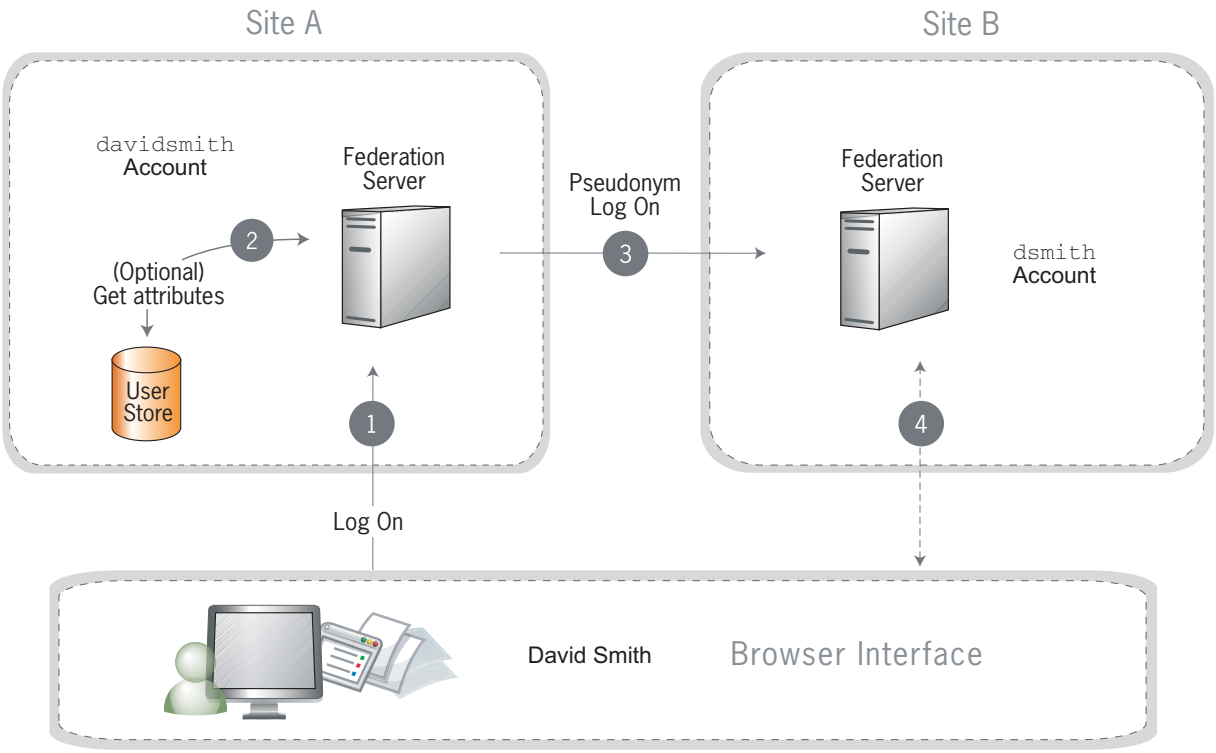

Figure 20: Account Linking

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 http://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).

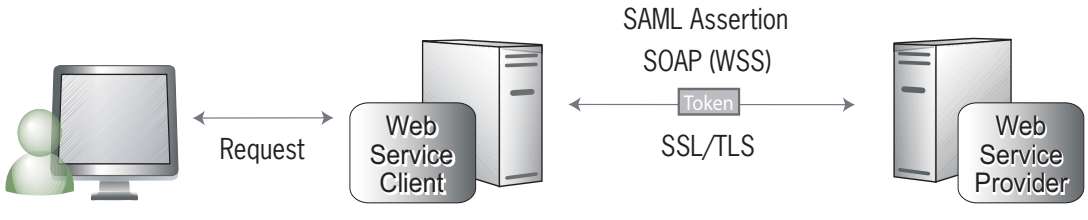

Figure 21: WSS Token Transfer

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 55).

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.

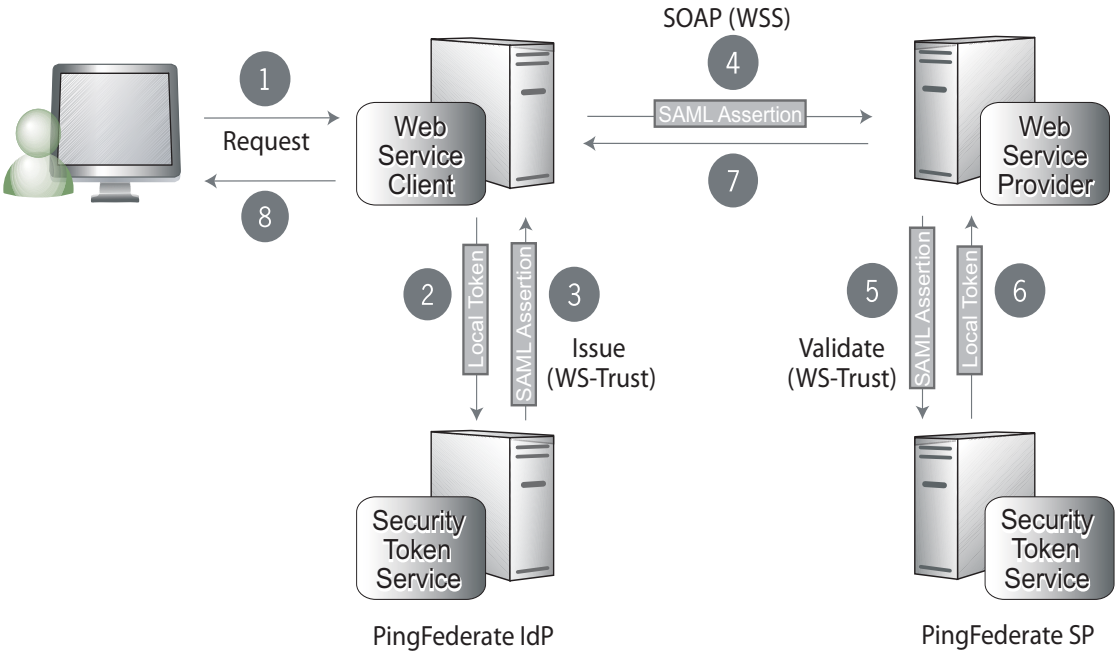

Figure 22: Token Exchange (Example)

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.

Transport and Message Security

SAML defines two main ways of securing its 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. 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* available on the SSTC Web site.

Appendix A

Using the SafeNet Luna HSM

Federal Information Processing Standard (FIPS) 140-2 requires the storage and processing of all keys and certificates on a certified cryptographic module. To meet this requirement, PingFederate is engineered and tested with the standard-compliant SafeNet Luna SA Hardware Security Module (HSM).

If your site requires FIPS 140-2 compliance, before running PingFederate you must install and configure the Luna SA HSM according to the manufacturer's documentation. Once this process is complete, follow the steps outlined below to configure PingFederate to interact with the HSM for key generation, storage, and operation.

Note that some restrictions exist in the operation of PingFederate when using an HSM:

- PingFederate must be running with JDK 5.
- Private encryption 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>/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" 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.

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 will run PingFederate.

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

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

   For Windows:

   Copy these files from the `Program Files\LunaSA\JSP\lib` folder into your `JAVA_HOME\jre\lib\ext` folder:

   - `LunaAPI.dll`
   - `LunaJCASP.jar`
   - `LunaJCESP.jar`

   For UNIX/Linux:

   Copy these files from the `/usr/lunasa/jsp/lib` directory into your `JAVA_HOME/jre/lib/ext` directory:

   - `libLunaAPI.so`
   - `LunaJCASP.jar`
   - `LunaJCESP.jar`

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

3. Install PingFederate on the network interconnected to the HSM (see "Installation" on page 11).

4. In the `<pf_install>/server/default/data` directory, delete files with the extension `.jks`, specifically:

   - `ping-dsig.jks`
   - `ping-ssl-server.jks`
   - `ping-ssl.jks`
   - `ping-trust.jks`

5. In the `hivemodule.xml` file in the `<pf_install>/server/default/conf/META-INF` directory, comment out this section of XML code (comment indicators are in **bold**):

```
<!-- Use this service-point if you are using default certificate
storage -->
<!--
    <service-point id="JCEManager"
interface="com.pingidentity.crypto.JCEManager">
         <invoke-factory>
             <construct
class="com.pingidentity.crypto.SunJCEManager"/>
         </invoke-factory>
     </service-point>
-->
```

6. Just below the code in the last step, uncomment this code:

```
<service-point id="JCEManager"
interface="com.pingidentity.crypto.JCEManager">
         <invoke-factory>
             <construct
class="com.pingidentity.crypto.LunaJCEManager"/>
         </invoke-factory>
  </service-point>
```

7. Later in the same file, comment out this section:

```
<!-- Use this service-point if you are using default certificate
storage -->
<!--
    <service-point id="CertificateService"
interface="com.pingidentity.crypto.CertificateService">
         <invoke-factory>
             <construct
class="com.pingidentity.crypto.CertificateServiceImpl"/>
         </invoke-factory>
     </service-point>
-->
```

8. Just below the code in the last step, uncomment this code:

```
<service-point id="CertificateService"
interface="com.pingidentity.crypto.CertificateService">
         <invoke-factory>
             <construct
class="com.pingidentity.crypto.LunaCertificateServiceImpl"/>
         </invoke-factory>
     </service-point>
```

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 at the bottom of this file from OFF to LUNA, as shown below:

```
#
# This property denotes FIPS mode. Current values are:
# LUNA - denotes a SafeNet implementation
# OFF - Use the default Sun keystore/JCE implementation
#
pf.hsm.mode=LUNA
```

11. Save and close the `run.properties` file.

12. From the `<pf_install>/bin` directory, run the `hsmpass.bat` batch file for Windows or the `hsmpass.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.

    Be aware that Java echoes each key press on the console; others might be able to see the password.

13. In your Java SDK directory, open the file `java.security` in the `jre/lib/security` directory and add the two lines in boldface to *top of the list* of security providers:

```
# List of providers and their preference orders (see above):
security.provider.1=com.chrysalisits.crypto.LunaJCAProvider
security.provider.2=com.chrysalisits.cryptox.LunaJCEProvider
```

    Renumber the existing security providers in the list accordingly. For example, a correctly configured list would look similar to this:

```
security.provider.1=com.chrysalisits.crypto.LunaJCAProvider
security.provider.2=com.chrysalisits.cryptox.LunaJCEProvider
security.provider.3=sun.security.provider.Sun
security.provider.4=sun.security.rsa.SunRsaSign
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
```

14. Save and close the `java.security` file.

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 15).

Index

A

account linking 51
administrative console
  accessing 15
  logging on using LDAP 16
  navigating 25
architecture
  enterprise 7
artifact
  artifact/artifact, SP-initiated SSO 43
  artifact/POST, SP-initiated SSO 39
  IdP-initiated SSO 34, 47
  POST/artifact, SP-initiated SSO 41
  redirect/artifact, SP-initiated SSO 42
  use 31
assertions
  about 30
attributes
  masking in log files 48

B

bindings
  artifact 31
  POST 31
  redirect 31
browser requirements 12
buttons, administrative console 26

C

configuration screens
  navigating 25
console buttons 26

D

data store
  compatibility 12
deployment diagrams 17

H

hardware requirements 13

I

IdP-initiated SLO 48
IdP-initiated SSO
  Artifact 34, 47
  POST 32, 46
installation
  federation server 14
  JDK 13
  Linux service 19
  Windows service 19

J

Java requirements 13
JDK installation 13

L

LDAP, using for authentication 16
Linux
  service 19

M

main menu 23
masking attributes in logs 48

N

navigation 23
  buttons 26

O

OASIS 29
operating systems 12

P

passive requestor profile (WS-Federation) 50
POST
  artifact/POST, SP-initiated SSO 39
  IdP-initiated SSO 32, 46
  POST/artifact, SP-initiated SSO 41
  POST/POST, SP-initiated SSO 36
  redirect/POST, SP-initiated SSO 38
  use 31
profiles 31
  IdP-initiated SSO
    Artifact 34, 47
    POST 32, 46
  SAML 1.x 32
  SAML 2.0 36
  SP-initiated SSO 35
    Artifact/Artifact 43
    Artifact/POST 39
    POST/Artifact 41
    POST/POST 36
    Redirect/Artifact 42
    Redirect/POST 38
pseudonyms
  using 51

R

redirect
  redirect/artifact, SP-initiated SSO 42
  redirect/POST, SP-initiated SSO 38
  use 31
requirements 12
  browser 12
  data store 12
  hardware 13
  Java 13
  operating systems 12
roles
  about 29

S

SAML
  account linking 51
  assertions 30
  bindings 31
  overview 29
  profiles, definition 31
  security 56
SAML 1.x
  profiles 32
SAML 2.0
  profiles 36
scalability 8
security
  SAML 56
service provider 30
session clean-up 48
single logout (SLO) 48
  session clean-up 48
single sign-on 36
SLO 48
SP-initiated SLO 48
SP-initiated SSO 35
  Artifact/Artifact 43
  Artifact/POST 39
  POST/Artifact 41
  POST/POST 36
  Redirect/Artifact 42
  Redirect/POST 38
starting and stopping the server
  Linux service 19
system navigation 23
system requirements 12

U

uninstalling 21

W

Web Services Security (WSS) 53
Windows service, installation 19
WS-Federation 50
WS-Trust
  description 54

WS-Trust STS
request types 54