CA SiteMinder®

Federation in Your Enterprise

r12.5

ca technologies

CA Technologies Product References

This document references the following CA Technologies products:

- CA SiteMinder®
- CA SiteMinder® SAML Affiliate Agent
- CA SiteMinder® Web Agent Option Pack
- CA SiteMinder® for Secure Proxy Server (CA SPS)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: SiteMinder Federation Deployments

This section contains the following topics:

Federation Deployment Models

CA SiteMinder Federation has two deployment models:

- Legacy Federation

  Legacy Federation (formerly Federation Security Services).

  Legacy federation is based on configuring SiteMinder objects, such as affiliate domains, authentication schemes, and policies to protect federated resources.

- Partnership Federation

  Partnership federation is based on configuring federated partnerships. The partnership model does not require configuration of SiteMinder-specific objects, such as domains, realms, and policies.

  Both deployments provide user authentication data in the form of a SAML assertion. The entity that consumes the assertion uses the assertion to identify the user. Upon successful authentication, the consuming entity makes the requested resources available. The result is a seamless experience for the user.

Install the SiteMinder Policy Server, the Administrative UI, and the Web Agent Option Pack to use either model.

Note: Federation is separately licensed from SiteMinder.

Basic Federation Concepts

Read the following sections before implementing SiteMinder Federation for information about some basic federation security concepts.

Note: Federation is separately licensed from SiteMinder.

Federation Specifications

SiteMinder supports the following federation specifications:

Security Assertion Markup Language (SAML)

The Security Assertion Markup Language (SAML) is a standard from the Organization for the Advancement of Structured Information Standards (OASIS). This industry standard defines an XML framework for exchanging authentication and authorization information.

SAML defines assertions as a means to pass security information about users between entities. SAML assertions are XML documents that contain information about a specific subject, such as a user. An assertion can contain several different internal statements about authentication, authorization, and attributes.

SAML defines two browser-based protocols that specify how SAML assertions are passed between partners to facilitate single sign-on.

The profiles are:

- Browser/artifact profile—defines a SAML artifact as a reference to a SAML assertion.
- Browser/POST profile—returns a response that contains an assertion.

Note: For SAML 2.0, the artifact and POST profiles are referred to as HTTP bindings.

For SAML specifications and information about SAML profiles, refer to the Organization for the Advancement of Structured Information Standards (Oasis).

SiteMinder supports the following SAML standards and profiles:

- SAML 1.0 Artifact profile only (legacy federation only)
- SAML 1.1 Artifact and POST profile
- SAML 2.0 Artifact and POST profile

WS-Federation (legacy federation only)

Active Directory Federation Services (ADFS) is web services-based solution from Microsoft for federated single sign-on (SSO). ADFS runs on a Windows server and accomplishes SSO by letting partners securely share user identity information and access rights across a secure network. ADFS extends SSO functionality to internet applications, letting users have a seamless web SSO interaction when they access web-based applications of the organization.

ADFS uses the following specifications:

- Web Services Federation (WS-Federation)
- WS-Federation Passive Requestor Profile (WS-F PRP)
- WS-Federation Passive Requestor Interoperability Profile

For WS specifications and background documentation, and information about ADFS profiles, go to the Microsoft website.

Note: This profile is only available with SiteMinder Legacy Federation.

Entities in a Federated Network

In a federated network, one entity generates SAML assertions. Assertions contain information about a user whose identity is maintained locally at the site that generates them. The other entity uses the SAML assertions to authenticate a user and to establish a session for the user.

Depending on the protocol, these two entities are named differently, but the functions they serve are the same.

Protocol	Generates Assertions	Consumes Assertions
SAML 1.0 and 1.1	Producer	Consumer
SAML 2.0	Identity Provider (IdP)	Service Provider (SP)
WS-Federation	Account Partner (AP)	Resource Partner (RP)

A site can be the asserting party (producer/IdP/AP) and the relying party (consumer/SP/RP).

Note: The partnership federation model only supports SAML 1.1 and SAML 2.0.

Chapter 2: Federation Deployment Considerations

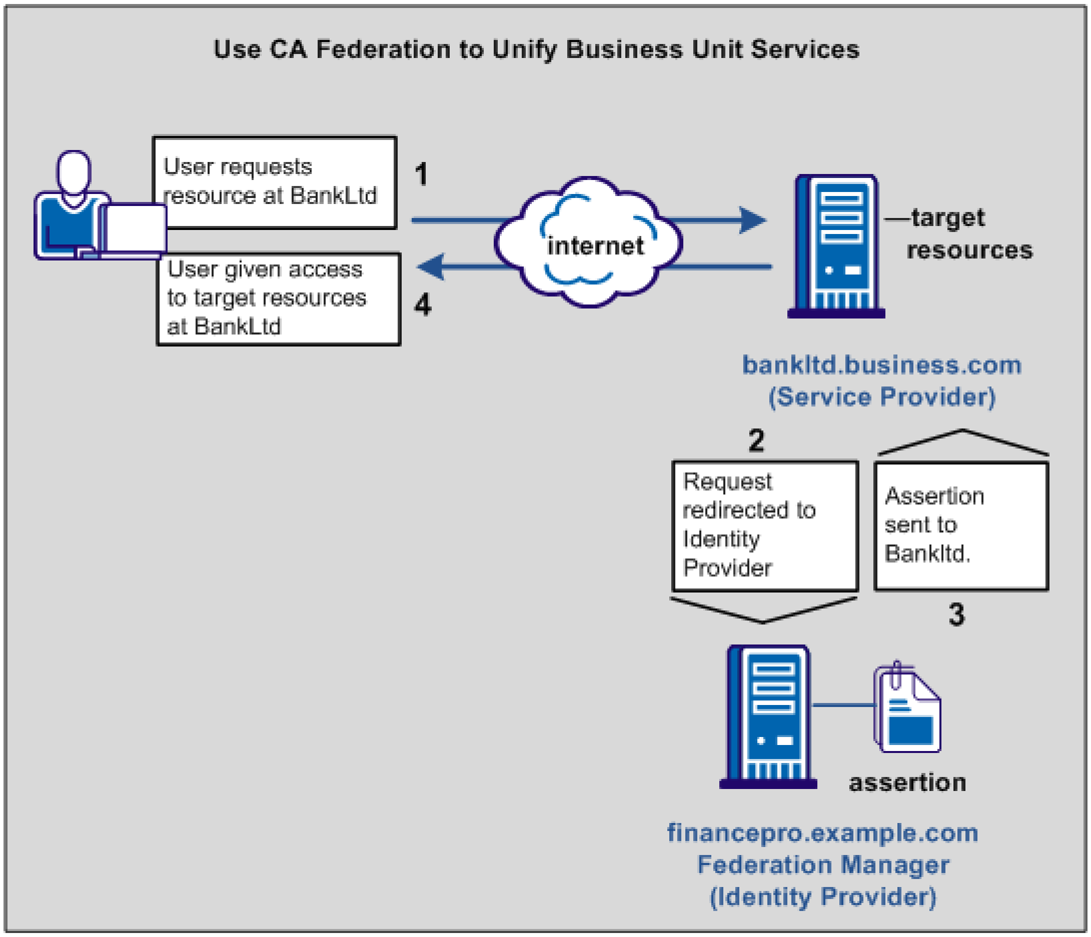
Federation Business Case

A sample business case best illustrates how SiteMinder federation can solve a common business problem.

In this business case, Financepro is a financial planning firm that recently bought the banking firm BankLtd to provide private banking to its clients. These two companies have different information infrastructures, but they want to appear as one company to their customers. To solve this problem, they set up a federated partnership.

By establishing a federated relationship, the two companies can provide a seamless customer experience using single sign-on. Customers can travel between Financepro and BankLtd without constantly being challenged to authenticate. Additionally, the sharing of customer identities and customer information can further customize the user experience and cross-promote the financial products of each partner.

The following illustration shows the federated partnership between Financepro and BankLtd. The flow of communication is based on a SAML 2.0 Service Provider-initiated single sign-on.


Use CA Federation to Unify Business Unit Services
User requests resource at BankLtd
1
User given access to target resources at BankLtd
4
internet
—target resources
bankltd.business.com (Service Provider)
2
Request redirected to Identity Provider
Assertion sent to Bankltd.
3
assertion
financepro.example.com Federation Manager (Identity Provider)

The illustration describes the following information flow:

1. The user tries to access a federated resource at BankLtd.
2. The user is redirected to the Financepro for authentication and the assertion is generated.
3. The assertion is passed back to BankLtd.
4. Single sign-on occurs based on either a SAML HTTP-Artifact or HTTP-POST. The user gets access to the target resource.

For this partnership to work, decide how the partnership functions before implementing the relationship using Federation Manager.

The issues to consider include:

- How users are identified across the partnership.
- What attributes get sent in an assertion and for what purpose.
- Which federation binding to use (SAML or WS-Federation)

Your decisions help structure the business partnership.

User Identification Across the Partnership

Business partners have their own method of defining user identity in their respective user stores. How users are identified determines how one partner can map its users to the other partner.

Consider the following scenarios:

- The User ID is the same at the user store of each site.

  Account linking is the method of user identification.
- The User ID is unique at the user store of each site.

  Identity mapping is the method of user identification. At FinancePro, a customer is identified as JohnDoe, while at BankLtd this same customer is identified as DoeJ. The partners must agree on a user attribute profile to use for identity mapping.
- The User ID does not exist at the relying party.

  Account provisioning is the method for user identification. Provisioning an account can require creating an account for a user or simply populating an existing user account with information in the SAML assertion.

The user identification decision determines what information is sent as the user identity in the assertion.

User Mapping

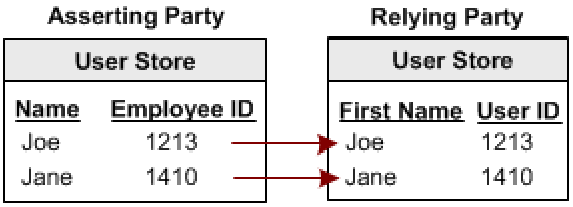
User mapping is the ability to establish a relationship between a user identity at one business and a user identity at another business. Map remote users at the asserting party to local users at the relying party.

The types of mapping are as follows:

- One-to-one mapping maps a unique remote user directory entry at the producing authority to a unique user entry at the consuming authority.

  One-to-one mapping, or account linking, links an account at the asserting party to an account at the relying party.

  The following illustration shows one-to-one mapping.



Asserting Party

User Store

Name	Employee ID
Joe	1213
Jane	1410

Relying Party

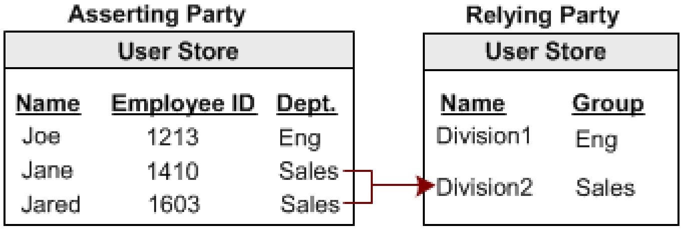
User Store

First Name	User ID
Joe	1213
Jane	1410

- N-to-one mapping maps a group of remote user directory entries to a single local profile entry.

  N-to-one mapping allows several user records at a producing authority to be mapped to one user record or profile at a consuming authority. An administrator at the relying party can use n-to-one mapping for a group of remote users without maintaining a record for each remote user.

  The following illustration shows n-to-one mapping:



Asserting Party

User Store

Name	Employee ID	Dept.
Joe	1213	Eng
Jane	1410	Sales
Jared	1603	Sales

Relying Party

User Store

Name	Group
Division1	Eng
Division2	Sales

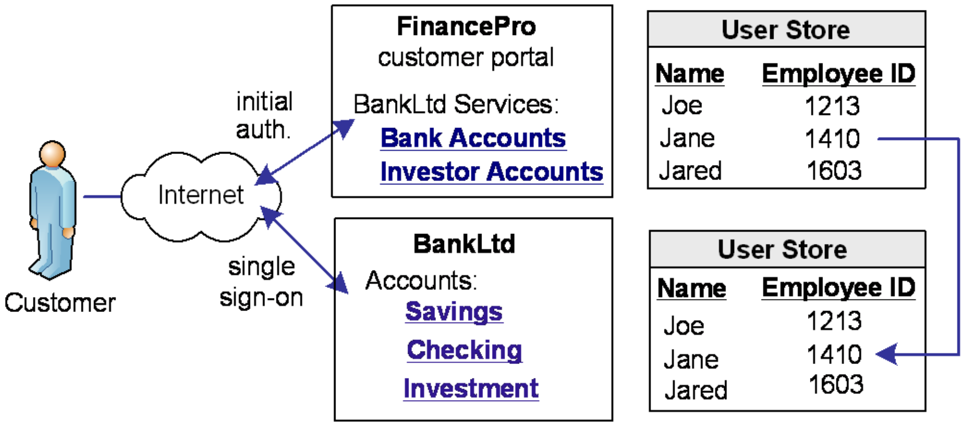
For legacy federation, user mapping is configured as part of a federation authentication scheme. For partnership federation, user mapping is configured as part of the name ID and attribute settings.

Account Linking to Establish a Federated Identity

When a customer at FinancePro accesses a resource at BankLtd, the NameID is always in the assertion. This identifier allows BankLtd to determine who the customer is and the level of access to allow for that customer.

The NameID can establish a federated identity when the user store at each partner identifies the users in the same way with the same ID.

The following figure shows the user store at each site with the same employee IDs.



User Store

Name	Employee ID
Joe	1213
Jane	1410
Jared	1603

User Store

Name	Employee ID
Joe	1213
Jane	1410
Jared	1603

Federation Manager lets you configure account linking as part of the partnership configuration process. You specify a NameID format and Name ID type, which determines the type of value that defines the Name. You associate the specific Name ID type, with a static, user, or DN attribute from a user directory. The NameID that Federation Manager includes in the assertion conforms to the configuration you define.

When the relying party receives the assertion, the user disambiguation process at BankLtd occurs. The process links the NameID value in the assertion to a record in its user store.

Identity Mapping to Establish a Federated Identity

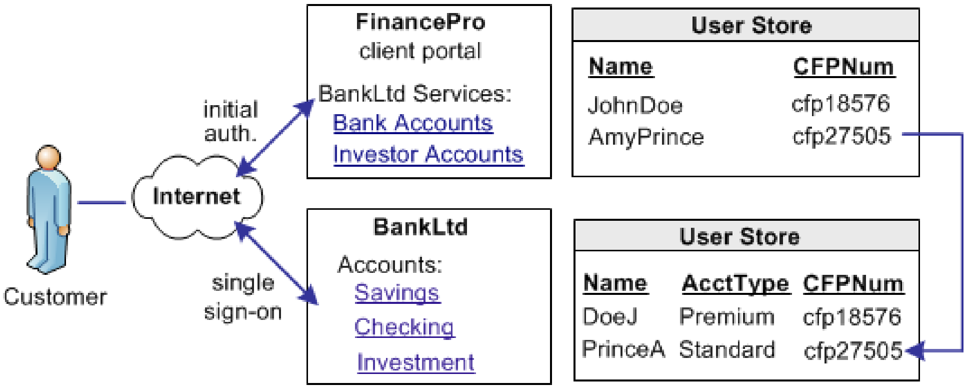
An investor at Financepro authenticates and selects a link to access information at BankLtd. The investor is taken directly to the accounts area of the BankLtd website without having to sign on.

BankLtd maintains user identities for all customers at Financepro, but the identities differ from the identities at FinancePro. For example, at FinancePro, JohnDoe is a customer. At BankLtd, this same customer is identified as DoeJ. Regardless, BankLtd must control access to sensitive portions of the company website. To establish the federated identity, the partners agree on an attribute that maps to the appropriate identity for a single customer at either site.

The partners agree on which attribute to use during an out-of-band exchange of information, meaning that the agreement is not part of any communication in any message over a channel. For this example, the attribute that the partners agree upon is a certified financial planner license number, referred to as the CFPNum in each user store.

When a customer tries accessing the federated resource at BankLtd, the request triggers the single sign-on process. The assertion that is generated at FinancePro contains the CFPNum attribute. When BankLtd receives the assertion, an application at its site has to perform the user disambiguation process. The process relies on the attribute to determine which profile identity is used for the request.

The following illustration shows how the same users are identified differently at each partner.



User Store

Name	CFPNum
JohnDoe	cfp18576
AmyPrince	cfp27505

User Store

Name	AcctType	CFPNum
DoeJ	Premium	cfp18576
PrinceA	Standard	cfp27505

Federation Manager lets you configure identity mapping as part of the partnership configuration process. For the NameID and attribute configuration, you define an attribute called CFPID. Associate this attribute with the user attribute CFPNum, which is the name of the attribute in the user store at each partner.

Federation Manager includes the attribute in the assertion. When BankLtd receives the assertion, the user disambiguation process links the attribute in the assertion to the appropriate record in its user store.

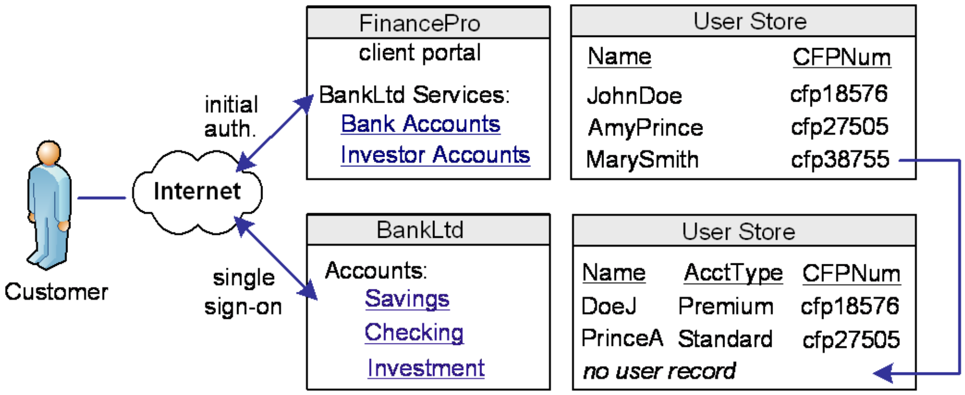
User Provisioning to Establish a Federated Identity (partnership federation only)

Partnership federation can work with provisioning applications at the relying party to establish an identity.

A client at Financepro, Mary Smith, authenticates and clicks a link to access information at BankLtd. Initially, BankLtd cannot find a user account for Mary Smith. BankLtd wants to protect sensitive portions of its website while allowing new customers.

BankLtd has configured Federation Manager to implement provisioning to establish the new federated identity for Mary Smith. Federation Manager redirects Mary Smith to the provisioning server at BankLtd. The provisioning application, using identity information from Federation Manager, creates a user account in the user store.

The following illustration shows the user stores at FinancePro and BankLtd.



Federation Manager lets you configure provisioning as part of the partnership configuration at the relying party. In this example, you select remote provisioning and determine how assertion data is delivered to the BankLtd provisioning server. This configuration enables the dynamic creation of a user entry in the user store.

Attributes for Customizing an Application

Federation Manager offers two ways of using attributes to customize target applications.

Attributes Added to Assertions at the Asserting Party

You can include attributes from a user store record in an assertion to identify a user for the purpose of customizing an application.

Servlets, web applications, and other custom applications can use attributes to display customized content or enable and disable other custom features. When used with web applications, attributes can implement fine-grained access control by limiting user activity at the target site. For example, you send an attribute variable named Account Balance and set it to reflect the account holdings of the user at BankLtd.

Attributes take the form of name/value pairs. When the relying party receives the assertion, it makes the attribute values available to applications.

Attribute Mapping at the Relying Party

The relying party receives a set of assertion attributes, which can be mapped to a set of application attributes being delivered to the target application.

For example, FinancePro includes an assertion attribute CellNo=5555555555. At BankLtd, this attribute name is transformed to an application attribute Mobile=5555555555. The attribute name is converted but the value remains the same.

Multiple assertion attributes can also be transformed into a single application attribute. For example, FinancePro sends an incoming assertion with the attributes Acct=Savings and Type=Retirement and transformed at BankLtd into FundType= Retirement Savings.

Federation Profile for Single Sign-on

Determining the profile for a partnership depends on the binding that each side can support.

For a new federation, there are no legacy requirements for either partner. Therefore, the recommended profile to use for single sign-on is SAML 2.0 POST profile. SAML 2.0 POST profile offers secure transmission of assertion data and the configuration process is simpler than SAML Artifact profile. If, however, the agreement of two partners requires SAML Artifact, this binding can also be implemented.

For legacy federation deployments that use Active Directory Federation Services (ADFS), configure WS-Federation.

Federating with Each Federation Manager Model

The legacy federation or partnership federation model can establish a federated partnership between Financepro and BankLtd. Using federation, users move between each company as if they are one company.

Partnership Federation Model

Configure the partnership model in the Administrative UI, guided by a partnership wizard. The partnership objects focus on creating partnerships and identifying each side of the partnership to accomplish single sign-on.

These steps include:

1. Configuring a Partnership

   Names the partnership and identifies the two entities that make up the partnership.

2. Establishing the Federation Users/User Identification

   Specifies the users for which the asserting party generates assertions and the relying party authenticates.

3. NameID and Attributes

   Determines how a federated identity is established and lets you add attributes to identify and customize the content of the assertion.

   Using NameID and attributes, you can verify that the appropriate information is available to the application at the relying party. The NameID and Attributes step is where you configure account linking and identity mapping.

4. SSO and Single Logout

   Defines Single Sign-on (Artifact or POST binding), including the location of the service consuming assertions at the relying party. For SAML 2.0, you can configure more features, such as single logout (SLO), Enhanced Client or Proxy (ECP) profile, and Identity Provider Discovery profile.

5. Signature and Encryption

   Defines the signature and encryption options for secure exchange of data, including:

   - assertions
   - authentication requests
   - SAML 2.0 single logout requests and responses

6. Application Integration

   Enables you to configure redirection to the target application, lets you set up provisioning of user records, and define relying-party side attribute mapping. You can also set up redirects for failed user authentication.

Legacy Federation Model

The legacy federation model focuses on the domain, realm, rule, authentication schemes, and policy objects.

If SiteMinder is the asserting party, the configuration steps include:

1. Configuring an entity in an affiliate domain

   Names the partner for which the asserting party generates assertions.

2. Establishing federation users

   Specifies the user directories for which the asserting party generates assertions and the relying party authenticates.

3. Selecting profiles (SAML or WS-Federation) for transactions

   Determines how a federated identity is established. In the profiles configuration, you add attributes to identify and customize the content of the assertion.

   Using NameID and attributes, you can verify that the appropriate information is available to the application at the relying party. The profiles configuration is where you specify account linking and identity mapping.

   As part of the profiles, configure single sign-on. For SAML 2.0, you can configure more features, such as single logout (SLO), Enhanced Client or Proxy (ECP) profile, and Identity Provider Discovery profile. For WS-Federation, you can configure sign-out.

4. Signature processing and encryption (SAML 2.0)

   Defines the signature options for secure exchange of assertions, authentication requests, and single logout requests and responses.

If SiteMinder is the relying party, the configuration steps include:

1. Setting up SAML and WS-Federation authentication schemes

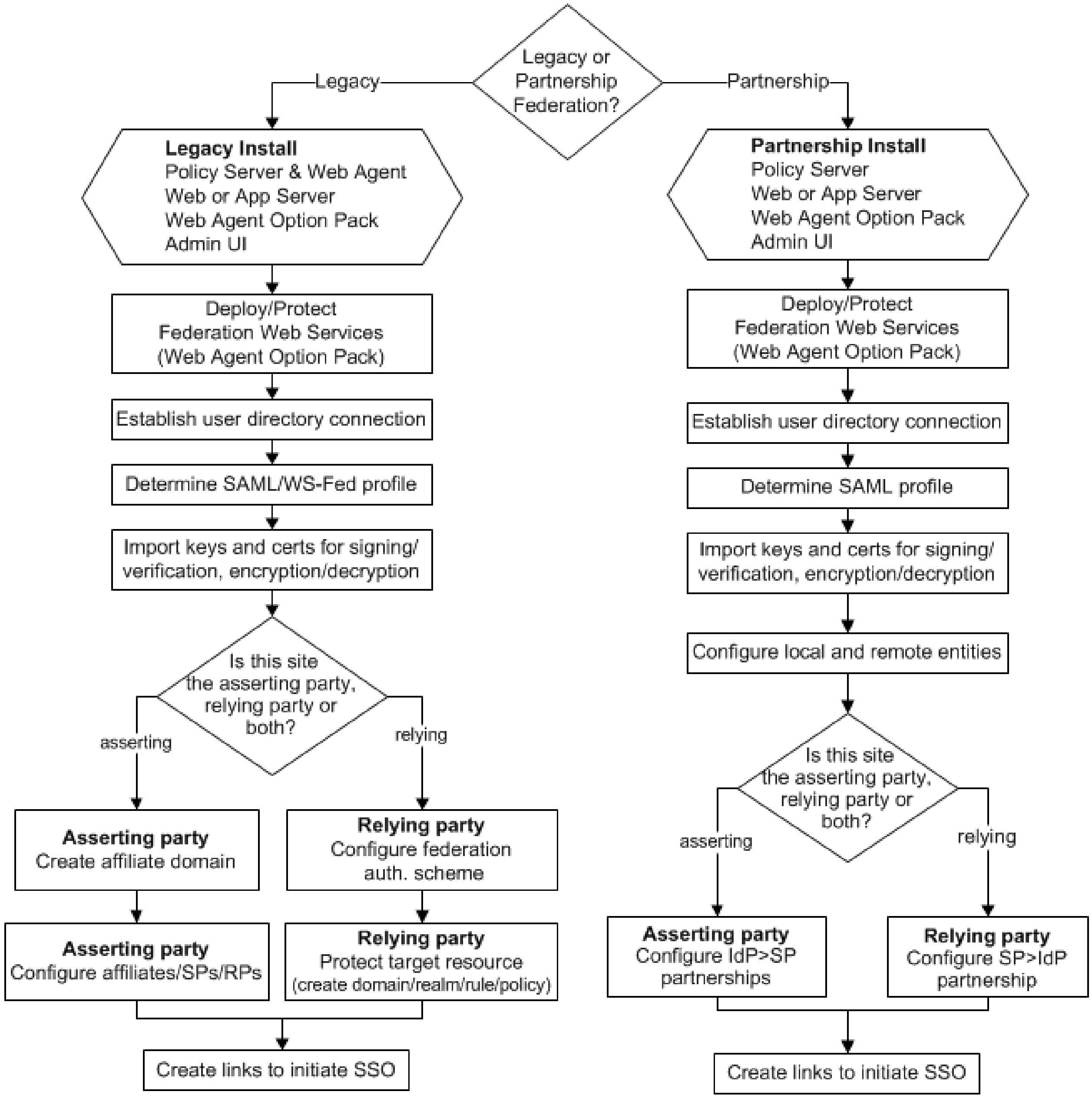
   Enables you to configure redirection to the target application, lets you set up provisioning of user records, and define relying-party side attribute mapping.

2. Configuring federation-specific settings included with the authentication scheme, such as single sign-on, single logout, sign-out, encryption, and decryption.

Federation Flow Diagram

Configure the components to establish successful federated partnerships. Most of these components are configurable using the Administrative UI.

The following flow chart highlights the general process for legacy federation and partnership federation.


Legacy or Partnership Federation?
Legacy
Partnership
Legacy Install
Policy Server & Web Agent
Web or App Server
Web Agent Option Pack
Admin UI
Deploy/Protect Federation Web Services (Web Agent Option Pack)
Establish user directory connection
Determine SAML/WS-Fed profile
Import keys and certs for signing/ verification, encryption/decryption
Is this site the asserting party, relying party or both?
asserting
relying
Asserting party
Create affiliate domain
Relying party
Configure federation auth. scheme
Asserting party
Configure affiliates/SPs/RPs
Relying party
Protect target resource (create domain/realm/rule/policy)
Create links to initiate SSO
Partnership Install
Policy Server
Web or App Server
Web Agent Option Pack
Admin UI
Deploy/Protect Federation Web Services (Web Agent Option Pack)
Establish user directory connection
Determine SAML profile
Import keys and certs for signing/ verification, encryption/decryption
Configure local and remote entities
Is this site the asserting party, relying party or both?
asserting
relying
Asserting party
Configure IdP>SP partnerships
Relying party
Configure SP>IdP partnership
Create links to initiate SSO

See the following guides for detailed instructions on required components and configuration procedures:

Legacy federation

Federation Manager Guide: Legacy Federation

Legacy federation refers to the product known as Federation Security Services.

Partnership federation

Federation Manager Guide: Partnership Federation

Partnership Federation refers to partnership model of federation, previously available only with the standalone Federation Manager product.

Chapter 3: Comparing Federation and Web Access Management for Single Sign-on

This section contains the following topics:

Advantages of Federation and Web Access Management

Federation and web access management (WAM) offer different benefits for single sign-on. Determining when to use federation or WAM single sign-on is dependent on your deployment.

Federation allows you to expand on your WAM capabilities; it does not replace those capabilities.

Federation has the following advantages:

- Many applications can handle federation directly out-of-the-box, such as SAP, SharePoint, WebLogic. These applications accept assertions.
- A direct connection to a centralized server is unnecessary. A federation request always goes through the asserting party to get the generated assertion. After a user gains access to content on one server, the user returns to the federation hub and gets redirected to the next server. Only if the user session times out at the hub does the user have to reauthenticate.
- Two models of SiteMinder federation. Partnership federation is business-centric, emphasizing relationships with partners. Legacy federation is protocol-centric and more customizable to the protocol specification.

These advantages make federated partnerships better for an environment where sites are remote, inaccessible, or under third-party control.

SiteMinder WAM single sign-on has the following advantages:

- Transactions are faster because there are fewer browser redirects.
- SiteMinder provides centralized authorization and auditing.
- Direct links can exist from one web server to another in a network without the user going through a centralized hub for assertion generation.

- SiteMinder offers timeout management.
- Applications are independent of a remotely initiated transaction.

These advantages make WAM single sign-on better suited to an environment with sites that are under your control, such as internal data centers.

Deployments that Favor Federation

Federation is advantageous in networks where your company does not control the server. For example, a third party owns the web server and does not allow you to install a web agent on the server. Also, when a remote server is in a location where there is a high network latency between the web agent and Policy Server. When you have no control over the target server, a SAML assertion is an ideal way to pass identity information.

The partners in a federated network follow the specific standards for the protocol used in communications. The common standards make the generating and consuming of assertions universal. The result is that the vendor at the asserting or relying party is not important nor is the remote location of each vendor.

Finally, federation is a good solution when timeouts are not a major concern, and obtaining identity information is the goal. External authorization checking is not a focus of federation.

Deployments that Favor Web Access Management

WAM single sign-on works best in an environment where you have control over each website. Having SiteMinder in the same data center as the website or other internal single sign-on environments are good deployments for web access management. Controlling over each website is also important for auditing your network performance and monitoring timeout issues.

WAM single sign-on lets you integrate with an application by way of a WAM session. WAM implementations also reduce some of the performance issues inherent with federation. For example, a transaction that is initiated by an asserting party can require several redirects after a user selects a link to make a request.

Chapter 4: Federation Web Services

This section contains the following topics:

Federation Web Services Overview

The Federation Web Services (FWS) application is installed with the Web Agent Option Pack on a server that has a connection to a SiteMinder Policy Server. The Federation Web Services and the SiteMinder Web Agent support the following web browser single sign-on profiles. These profiles convey information from one site to another through a standard browser.

The supported profiles are:

- SAML artifact profile 1.0 (legacy federation only)
- SAML artifact profile 1.1 and 2.0 (legacy federation and partnership federation)
- SAML POST profile 1.x and 2.0 (legacy federation and partnership federation)
- WS-Federation Passive Requestor profile (legacy federation only)

SAML 1.x Artifact and POST Profiles

For the SAML 1.x artifact and POST profiles, the Federation Web Services application uses the following services:

Assertion Retrieval Service (SAML 1.x Artifact only)

A producer-side component. This service handles a SAML request for the assertion that corresponds to a SAML artifact by retrieving the assertion from the SiteMinder session store. The SAML specification defines the assertion retrieval request and response behavior.

Note: Only the SAML artifact profile uses the assertion retrieval service.

Session Synchronization (SAML 1.x)

A producer-side component that validates and terminates sessions for the SAML Affiliate Agent (A SiteMinder value-added service, which uses a standards-based SOAP RPC mechanism).

Notification Alert (SAML 1.x)

A producer-side component that logs resource access notification events for the SAML Affiliate Agent (A SiteMinder value-added service, which uses a standards-based SOAP RPC mechanism).

SAML Credential Collector (SAML 1.x)

A consumer-side component that receives a SAML artifact or an HTTP form with an embedded SAML response and obtains the corresponding SAML assertion. The credential collector issues SiteMinder cookies to a browser of the user.

Intersite Transfer Service (SAML 1.x)

A producer-side component for the SAML POST profile. The intersite transfer service transfers a user from the producer site to a consumer site. For the SAML artifact profile, the Web Agent performs the same function as the intersite transfer service.

SAML 2.0 Artifact and POST Profiles

For SAML 2.0 artifact and POST profiles, the Federation Web Services application uses the following services:

Artifact Resolution Service (SAML 2.0 Artifact only)

An Identity Provider-side service that corresponds to the SAML 2.0 authentication using the HTTP-artifact binding. This service retrieves the assertion stored in the SiteMinder session store at the Identity Provider.

Note: Only the HTTP-artifact binding uses the artifact resolution service.

Assertion Consumer Service (SAML 2.0)

A Service Provider component that receives a SAML artifact or an HTTP form with an embedded SAML response and obtains the corresponding SAML assertion. The Assertion Consumer Service issues SiteMinder cookies to a browser.

Note: The Assertion Consumer Service accepts an AuthnRequest with an AssertionConsumerServiceIndex value of 0. All other values for this setting are denied.

AuthnRequest Service (SAML 2.0)

This service is deployed for use by SAML 2.0. A Service Provider can generate an <AuthnRequest> message to authenticate a user for cross-domain single sign-on. This message contains information that enables the Federation Web Services application to redirect the browser to the single sign-on service at the Identity Provider. The AuthnRequest service is used for POST and Artifact single sign-on.

Single Sign-on Service (SAML 2.0)

The single sign-on service enables an Identity Provider to process AuthnRequest messages. The service also invokes the assertion generator to create an assertion that is sent to the Service Provider.

Single Logout Service (SAML 2.0)

This service implements processing of single logout functionality, which an Identity Provider or a Service Provider can initiate.

Identity Provider Discovery Service (SAML 2.0)

Implements SAML 2.0 Identity Provider Discovery Profile and sets and retrieves the common domain cookie. An IdP requests to set the common domain cookie after authenticating a principal. An SP requests to obtain the common domain cookie to discover which Identity Provider a principal is using.

WS-Federation Passive Requestor Profile

For the WS-Federation Passive Requestor profile, the Federation Web Services application uses the following services:

Note: WS-Federation is only available with legacy federation.

Security Token Consumer Service

A Resource Partner component that receives a security token and extracts the corresponding SAML assertion. The Security Token Consumer Service issues SiteMinder cookies to a browser.

Single Sign-on Service

Enables an Account Partner to process a wsignin message and gather the necessary Resource Partner information to authenticate the user. This service also invokes the assertion generator to create an assertion that is sent to the Resource Partner.

Signout Service

Implements processing of single logout functionality by way of a signout servlet. An Account Partner or a Resource Partner can initiate signout.

Chapter 5: Creating a Legacy Federation Configuration in the Partnership Model

No direct migration path from legacy federation to partnership federation exists. Reproducing your legacy federation configuration in the partnership federation model requires recreating the legacy entities and configuring partnerships.

Legacy and partnership objects do not share a one-to-one correspondence. In the legacy federation model, configuring federation involves the following tasks at each partner:

Asserting Party

- Configuring affiliate domains
- Identifying the relying parties in the affiliate domains (SAML 1.x affiliates, SAML 2.0 Service Providers) and configuring the communication with those relying parties.

  Note: ADFS/WS-Federation is only supported with legacy federation. You cannot create WS-Federation objects in the partnership model.

Relying Party

- Configuring authentication schemes that define the relying party.
- Within the authentication scheme, specifying how the relying party consumes an assertion and how the relying party redirects users to the target application.

In a partnership model, recreating a legacy configuration involves:

- Configuring asserting and relying party entities that represent the business partners.
- Defining partnerships between the entities.

The following table shows the relationship between legacy federation components and partnership federation components.

Legacy Components (Asserting Party)	Partnership Components (Asserting Party)
SAML 1.1 Affiliate	SAML 1.1 Producer-to-Consumer partnership *partnership federation does not support SAML 1.0.*
SAML 2.0 Service Provider	SAML2 IdP-to-SP partnership

Legacy Components (Relying Party)	**Partnership Components (Relying Party)**
Authentication Scheme: SAML Artifact or POST Template	SAML 1.1 Consumer-to-Producer partnership
Authentication Scheme: SAML 2.0 Template	SAML2 SP-to-IdP partnership

If you plan to recreate your legacy federation objects in the partnership model, pay attention to the following settings:

Active

(Affiliate/Service Provider Properties and SAML authentication scheme dialog for legacy federation). If SiteMinder is using the legacy federation configuration, confirm this check box is selected. If you recreate the legacy configuration in the partnership federation model with similar values for identity settings, such as source ID, clear this check box before activating the partnership federation object.

SiteMinder cannot work with a legacy and partnership configuration that use the same identity values or a name collision occurs.

Artifact Protection Type

(SSO settings for partnership federation). Defines how the back channel is protected for HTTP-Artifact single sign-on. The legacy option indicates that SiteMinder protects the back channel. The partnership option indicates that the federation component within SiteMinder protects the back channel.

If you recreate your legacy federation configuration in the partnership federation model, you can use the legacy method of protecting the back channel. The legacy option lets the configuration use the existing URL for the Assertion Retrieval Service (SAML 1.x) or Artifact Resolution Service (SAML 2.0). By selecting legacy as the option, SiteMinder accepts the request. You do not have to modify the URL. If the artifact service URL is from the legacy configuration but only the partnership option is selected for this setting, SiteMinder rejects the request.

Important! For the legacy federation option, enforce the policy that protects the artifact service. The artifact service is a component of the Federation Web Services. SiteMinder creates policies for Federation Web Services automatically. However, you are required to indicate which partnership is permitted access to the service that retrieves artifacts. For more information, refer to the *Federation Manager Guide: Partnership Federation.*

Options: Legacy, Partnership

Note: SiteMinder r12.5 ships with the Federation Security Services User Interface (FSS UI) and the Administrative UI. If you switch from the FSS UI to the Administrative UI for SiteMinder configuration, do not return to the FSS UI for any modifications to any configuration objects. Once you begin with the Administrative UI, continue to use the Administrative UI exclusively. If you return to the FSS UI after using the Administrative UI, objects in the policy store can impair the function of the SiteMinder Policy Server.

Appendix A: Legacy Federation Use Cases and Solutions

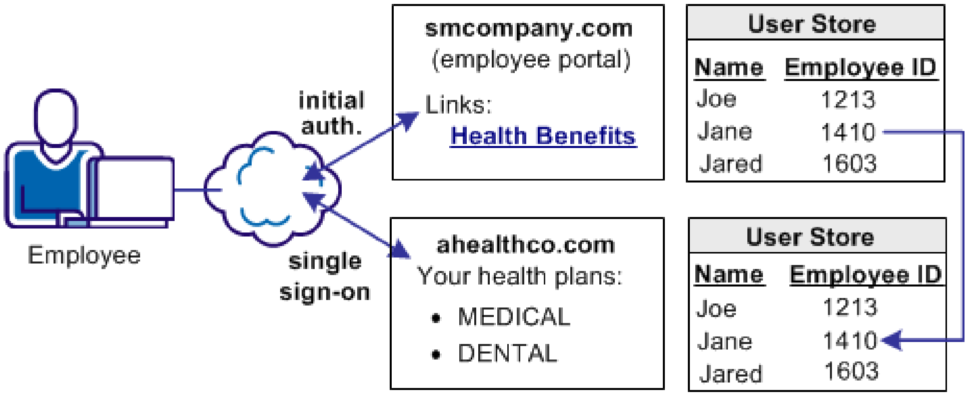
This section contains the following topics:

Use Case 1: Single Sign-on Based on Account Linking

In Use Case 1, smcompany.com contracts with a partner company, ahealthco.com to manage employee health benefits.

An employee of smcompany.com authenticates at an employee portal at his company’s site, www.smcompany.com and clicks a link to view her health benefits at ahealthco.com. The employee is taken to the ahealthco.com web site and is presented with her health benefit information without having to sign on to the website.

The following illustration shows this use case.



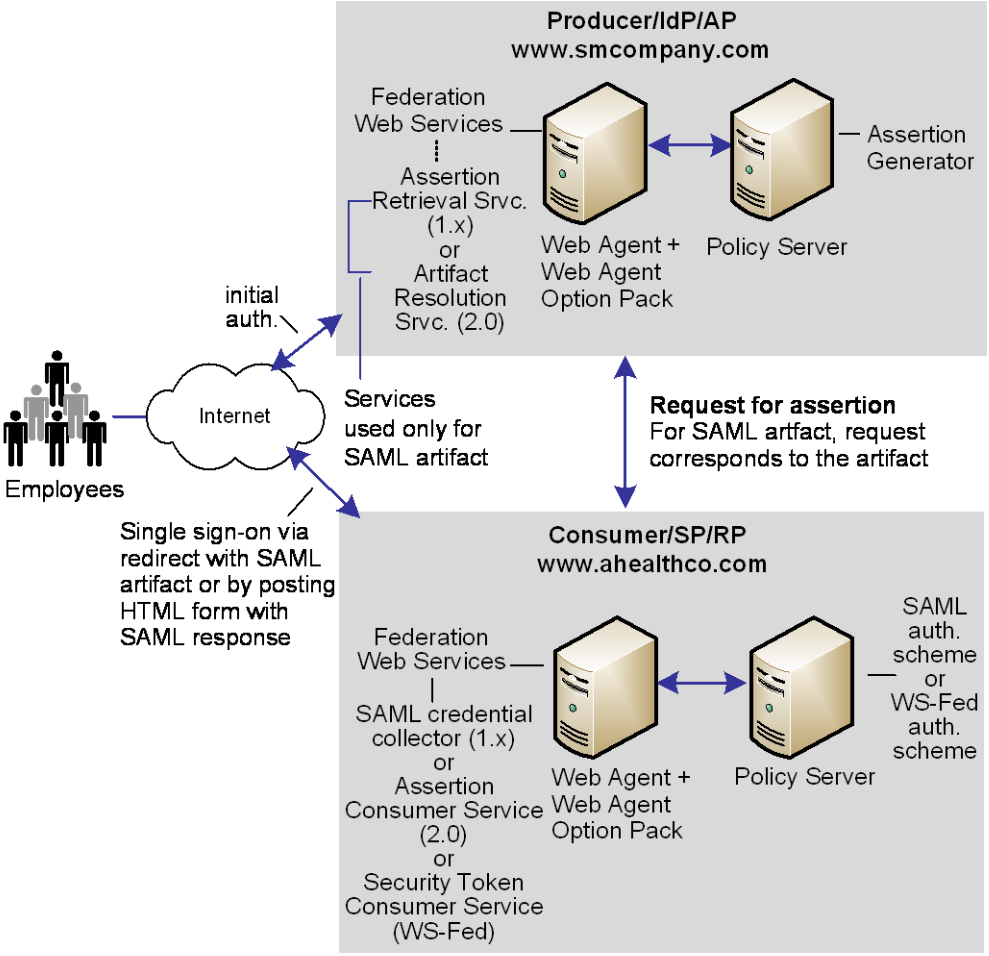
Name	Employee ID
Joe	1213
Jane	1410
Jared	1603

Name	Employee ID
Joe	1213
Jane	1410
Jared	1603

The company, ahealthco.com, maintains all health-related information for employees at smcompany.com. To do this, ahealthco.com maintains user identities for every employee of smcompany.com. When an employee of smcompany.com accesses ahealthco.com, an identifier for the employee is passed from smcompany.com to ahealthco.com in a secure manner. This identifier allows ahealthco.com to determine who the user is and the level of access to allow for that user.

Solution 1: Single Sign-on based on Account Linking

Solution 1 illustrates how legacy federation can be deployed at smcompany.com and ahealthco.com to solve Use Case 1: Single Sign-on Based on Account Linking (see page 33).


Producer/IdP/AP
www.smcompany.com
Federation Web Services
Assertion Retrieval Srvc. (1.x) or Artifact Resolution Srvc. (2.0)
Web Agent + Web Agent Option Pack
Policy Server
Assertion Generator
initial auth.
Internet
Employees
Services used only for SAML artifact
Request for assertion
For SAML artfact, request corresponds to the artifact
Single sign-on via redirect with SAML artifact or by posting HTML form with SAML response
Consumer/SP/RP
www.ahealthco.com
Federation Web Services
SAML credential collector (1.x) or Assertion Consumer Service (2.0) or Security Token Consumer Service (WS-Fed)
Web Agent + Web Agent Option Pack
Policy Server
SAML auth. scheme or WS-Fed auth. scheme

SiteMinder is deployed at both sites. The Web Agent with the Web Agent Option Pack are installed on a webserver system and the Policy Server is installed on another system. The installations are the same for smcompany.com and ahealthco.com.

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

Solution 1 Using SAML 1.x Artifact Authentication

In this example, smcompany.com is acting as the producer site. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the sequence of events is as follows:

1. The Web Agent provides the initial authentication.
2. When the employee clicks a link at smcompany.com to view the health benefits at ahealthco.com, the link makes a request to the Intersite Transfer Service at www.smcompany.com.
3. The Intersite Transfer Service calls the assertion generator, which creates a SAML assertion, puts the assertion into the SiteMinder session store. The service returns a SAML artifact.
4. The Web Agent redirects the user to www.ahealthco.com with the SAML artifact, in accordance with the SAML browser artifact protocol.

Ahealthco.com is acting as the consumer site. The SAML credential collector service handles the redirect request of the SAML artifact. The credential collector is part of the Federation Web Services application at ahealthco.com.

The sequence of events is as follows:

1. The SAML credential collector calls the SAML artifact authentication scheme to obtain the location of the assertion retrieval service at smcompany.com.
2. The SAML credential collector calls the assertion retrieval service at www.smcompany.com.
3. The assertion retrieval service at www.smcompany.com retrieves the assertion from the SiteMinder session store and returns it to the SAML credential collector at ahealthco.com.
4. The SAML credential collector then passes the assertion to the SAML artifact authentication scheme for validation and session creation. Also, it issues a SiteMinder session cookie to the browser.
5. The user is allowed access to resources at ahealthco.com based on the policies that are defined at the Policy Server at ahealthco.com. The Web Agent at ahealthco.com enforces the policies.

In this example, the administrator at smcompany.com configures an affiliate for ahealthco.com. The affiliate is configured with an attribute that is a unique ID for the user. This action causes the assertion generator to include that attribute as part of the user profile in a SAML assertion that is created for ahealthco.com.

The administrator at ahealthco.com configures a SAML artifact authentication scheme for smcompany.com. The authentication scheme specifies the location of the assertion retriever service at smcompany.com. The scheme also extracts the unique user ID from the SAML assertion, determines how to search the ahealthco.com user directory for the user record that matches the value from the assertion.

Solution 1 Using SAML 1.x POST Profile

In this example, smcompany.com is acting as the producer site. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the sequence of events is as follows:

1. The Web Agent provides the initial authentication.
2. When the employee clicks a link at www.smcompany.com to view the health benefits at ahealthco.com, the link makes a request to the Intersite Transfer Service at www.smcompany.com.
3. The Intersite Transfer Service calls the assertion generator, which creates a SAML assertion and signs the SAML response.
4. The signed response is then placed in an auto-POST HTML form and sent to the browser of the user.
5. The browser automatically POSTs a form to the Assertion Consumer URL (which is the SAML credential collector), at ahealthco.com. The form contains a SAML response as a form variable.

Ahealthco.com is acting as the consumer site. The SAML credential collector service handles the redirect request with the SAML response. The SAML credential collector that is part of the Federation Web Services at ahealthco.com.

The sequence of events is as follows:

1. The SAML credential collector calls for the requested target resource at ahealthco.com. The SAML POST profile authentication scheme protects the target resource.
2. Because the SAML POST profile scheme is protecting the resource, the SAML credential collector decodes the SAML response message.
3. Using the digitally signed SAML response message as credentials, the SAML credential collector calls the Policy Server at ahealthco.com.
4. The Policy Server verifies the signature and then authenticates the user using the SAML assertion embedded in the decoded SAML response message. Based on the assertion, the Policy Server lets the user log in.

5. After the user logs in, the SAML credential collector creates an SMSESSION cookie, places it in the browser of the user, and redirects the user to the target resource at ahealthco.com.
6. The user is allowed access to resources at ahealthco.com based on policies that are defined at the Policy Serve. The Web Agent enforces the policies.

In this example, the administrator at smcompany.com uses the UI to configure an affiliate object for ahealthco.com. The affiliate is configured with an attribute that is a unique ID for the user. This action causes the assertion generator to include that attribute as part of the user profile in a SAML assertion that is created for ahealthco.com.

The administrator at ahealthco.com configures a SAML POST profile authentication scheme for smcompany.com. The authentication scheme specifies how to extract the unique user ID from the SAML assertion. The scheme also defines how to search the user directory at ahealthco.com for the user record that matches the value from the assertion.

Solution 1 Using SAML 2.0 Artifact Authentication

In this example, smcompany.com is acting as the Identity Provider. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the sequence of events is as follows:

1. The Web Agent provides the initial authentication. When the user clicks a link at the Identity Provider, this action is referred to as an unsolicited response at the Identity Provider.
2. When the employee clicks a link at www.smcompany.com to view the health benefits at ahealthco.com, the link makes a request to the Single Sign-on Service at www.smcompany.com.
3. The single sign-on service calls the assertion generator creates a SAML assertion, stores the assertion in the SiteMinder session store, and returns a SAML artifact.
4. The Web Agent redirects the user to ahealthco.com with the SAML artifact, in accordance with the SAML browser artifact protocol.

Ahealthco.com is acting as the Service Provider. One of the components of the Service provider is the Assertion Consumer Service. The Assertion Consumer Service handles the redirect request containing the SAML artifact.

The sequence of events is as follows:

1. The Assertion Consumer Service calls the SAML 2.0 authentication scheme with HTTP-artifact binding to obtain the location of the artifact resolution service at smcompany.com.
2. The Assertion Consumer Service calls the artifact resolution service at www.smcompany.com.

3. The artifact resolution service at www.smcompany.com retrieves the assertion from the session store at smcompany.com and returns it to the artifact resolution service at ahealthco.com.
4. The Assertion Consumer Service passes the assertion to the SAML 2.0 authentication scheme for validation and session creation. The service issues a SiteMinder session cookie to the browser.
5. The user is allowed access to resources at ahealthco.com based on policies that are defined at the Policy Server at ahealthco.com. The Web Agent at ahealthco.com enforces the policies.

In this example, the administrator at smcompany.com configures a Service Provider object for ahealthco.com. The Service Provider is configured with an attribute that is a unique ID for the user. This action causes the assertion generator to include that attribute as part of the user profile in an assertion that is created for ahealthco.com.

The administrator at ahealthco.com configures a SAML 2.0 authentication scheme that uses the artifact binding for smcompany.com. The authentication scheme has the following information:

- The location of the artifact resolution service at smcompany.com.
- How to extract the unique user ID from the SAML assertion.
- How to search the user directory at ahealthco.com for the user record that matches the value in the assertion.

Solution 1 Using SAML 2.0 POST Binding

In this example, smcompany.com is acting as the Identity Provider. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the sequence of events is as follows:

1. The Web Agent provides the initial authentication. When the user clicks a link at the Identity Provider, this action is referred to as an unsolicited response at the Identity Provider.
2. When the employee clicks a link at www.smcompany.com to view the health benefits at ahealthco.com, the link makes a request to the Single Sign-on Service at www.smcompany.com.
3. The Single Sign-on Service passes calls the assertion generator, which creates a SAML assertion and signs the SAML response.
4. The signed response is then placed in an auto-POST HTML form and sent to the browser of the user.
5. The browser automatically POSTs a form to the Assertion Consumer URL at ahealthco.com. The form contains a SAML response as a form variable.

Ahealthco.com is acting as the Service Provider. The Assertion Consumer Service handles the redirect request with the SAML response. The service is part of the Federation Web Services at ahealthco.com.

The sequence of events is as follows:

1. The Assertion Consumer Service calls for the requested target resource at ahealthco.com. The SAML 2.0 authentication scheme protects this resource using the HTTP-POST binding.
2. Because the SAML 2.0 authentication scheme is protecting the resource, the Assertion Consumer Service passes the digitally signed SAML response message as credentials, to the Policy Server at ahealthco.com.
3. The Policy Server verifies the signature and then authenticates the user using the SAML assertion embedded in the decoded SAML response message. Based on the assertion, the Policy Server lets the user log in.
4. After the user logs in, the Assertion Consumer Service creates an SMSESSION cookie, places it in the browser of the user, and redirects the user to the target resource at ahealthco.com.
5. The user is allowed access to resources at ahealthco.com based on policies that are defined at the Policy Server. The Web Agent enforces the policies.

In this example, the administrator at smcompany.com configures a Service Provider object for ahealthco.com. The Service Provider is configured with an attribute that is a unique ID for the user. This action causes the assertion generator to include that attribute as part of the user profile in a SAML assertion that is created for ahealthco.com.

The administrator at ahealthco.com configures a SAML 2.0 authentication scheme with the HTTP-POST binding for smcompany.com. The authentication scheme has the following information:

- How to extract the unique user ID from the SAML assertion.
- How to search the user directory at ahealthco.com for the user record that matches the value from the assertion.

Solution 1 Using WS-Federation Passive Requestor Profile

In this example, smcompany.com is acting as the Account Partner. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the sequence of events is as follows:

1. The user visits an unprotected site selection page at ahealthco.com.
2. This link points to the Single Sign-on Service at the Account Partner, www.smcompany.com. The Web Agent provides the initial authentication.

3. The Single Sign-on Service calls the WS-Federation Assertion Generator, which creates a SAML 1.1 assertion. WS-Federation Assertion Generator signs the assertion and wraps the assertion in a security token response message.
4. The response is then placed in an auto-POST HTML form as a form variable and sent to the browser of the user.
5. The browser automatically POSTs a form to the Security Token Consumer Service URL at ahealthco.com.

Ahealthco.com is acting as the Resource Partner. The Security Token Consumer Service handles the redirect request with the SAML response. The service is part of the Federation Web Services application.

The sequence of events is as follows:

1. The Security Token Consumer Service calls for the requested target resource at ahealthco.com. The WS-Federation authentication scheme protects this resource.
2. Because the WS-Federation authentication scheme is protecting the resource, the Security Token Consumer Service passes the signed assertion as credentials to the Policy Server at ahealthco.com.
3. The Policy Server verifies the signature and then authenticates the user using the SAML assertion embedded in the decoded SAML response message. Based on the assertion, the Policy Server lets the user log in.
4. After the user logs in, the Security Token Consumer Service creates an SMSESSION cookie. The service then places the cookie in the browser and redirects the user to the target resource at ahealthco.com.
5. The user is allowed access to resources at ahealthco.com based on policies that are defined at the Policy Server. The Web Agent enforces the policies.

In this example, the administrator at smcompany.com configures a Resource Partner object for ahealthco.com. The Resource Partner is configured to include an attribute in the assertion that is a unique ID for the user. The assertion generator includes that attribute in the SAML assertion for ahealthco.com.

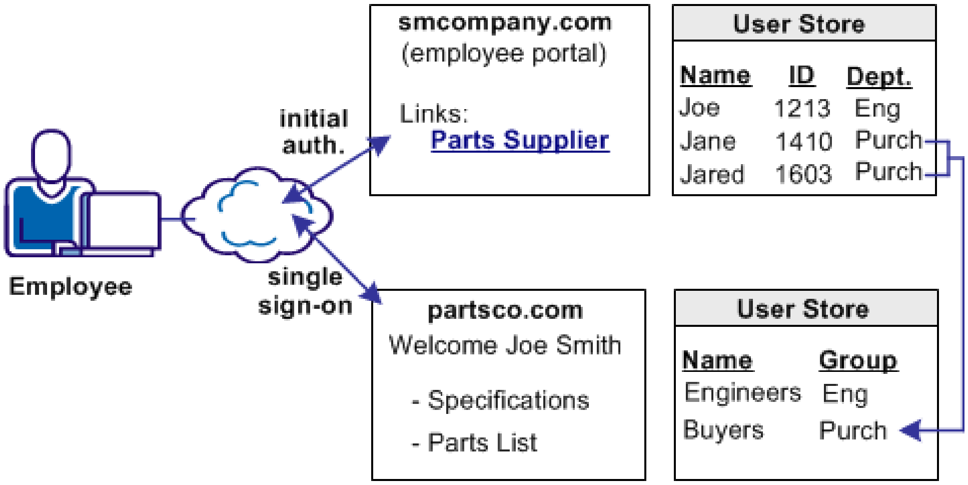
The administrator at ahealthco.com configures a WS-Federation authentication scheme for smcompany.com. The authentication scheme has the following information:

- How to extract the unique user ID from the SAML assertion.
- How to search the user directory at ahealthco.com for the user record that matches the value from the assertion.

Use Case 2: Single Sign-on Based on User Attribute Profiles

In Use Case 2, smcompany.com buys parts from a partner named partsco.com.

An engineer authenticates at the employee portal, smcompany.com and clicks a link to access information at partsco.com. Being an engineer at smcompany.com, the user is taken directly to the Specifications portion of the partsco.com website without having to log in.



User Store

Name	ID	Dept.
Joe	1213	Eng
Jane	1410	Purch
Jared	1603	Purch

User Store

Name	Group
Engineers	Eng
Buyers	Purch

When a buyer for smcompany.com authenticates and clicks a link for partsco.com, the buyer is taken directly to the Parts List portion of the partsco.com website. The buyer does not have to log in.

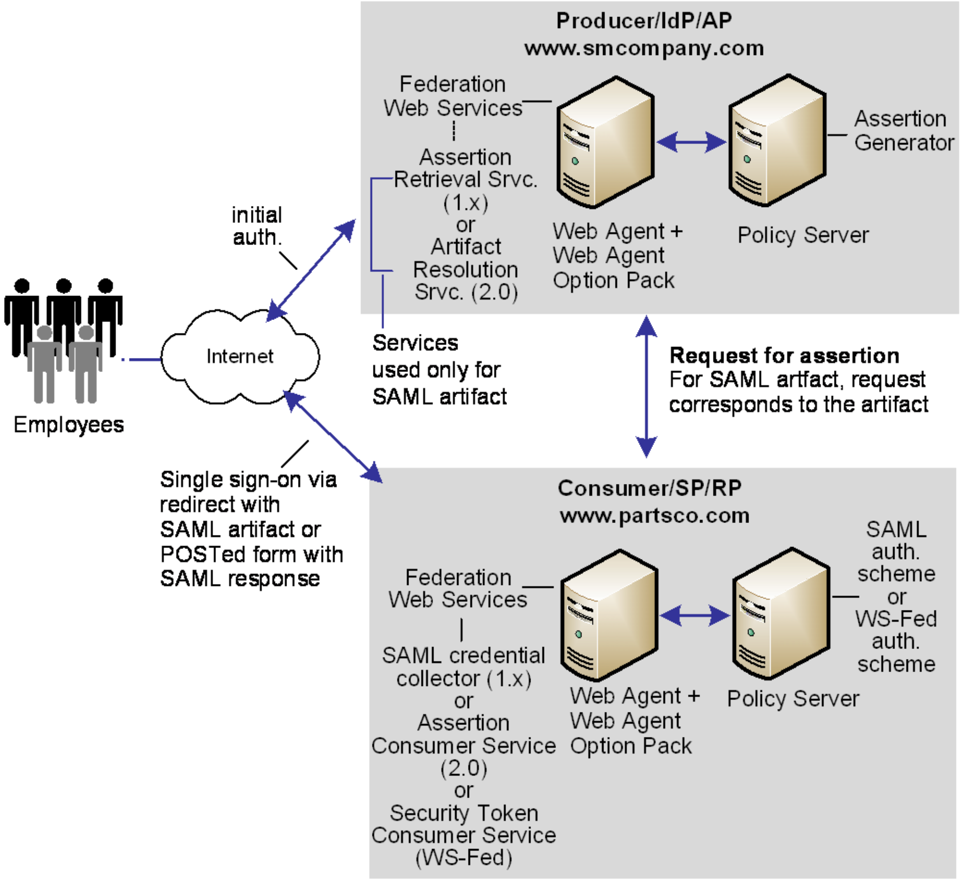
Additional attributes, such as the user name are passed from smcompany.com to partsco.com to personalize the interface for the individual user.

Partsco.com does not want to maintain user identities for all employees at smcompany.com, but the company wants to control access to sensitive portions of the website. To control the access, partsco.com maintains a limited number of profile identities for users at smcompany.com. One profile identity is maintained for engineers and one profile identity is maintained for buyers.

When an employee of smcompany.com accesses partsco.com, smcompany.com sends user attributes in a secure manner to partsco.com. Partsco.com uses the attributes to determine what profile identity controls access.

Solution 2: Single Sign-on based on User Attribute Profiles

Solution 2 shows how Federation Security Services can be deployed at smcompany.com and partsco.com to solve Use Case 2: Single Sign-on Based on User Attribute Profiles (see page 41).


Producer/IdP/AP
www.smcompany.com
Federation Web Services
Assertion Retrieval Srvc. (1.x) or Artifact Resolution Srvc. (2.0)
Web Agent + Web Agent Option Pack
Policy Server
Assertion Generator
initial auth.
Employees
Internet
Services used only for SAML artifact
Request for assertion
For SAML artfact, request corresponds to the artifact
Single sign-on via redirect with SAML artifact or POSTed form with SAML response
Consumer/SP/RP
www.partsco.com
Federation Web Services
SAML credential collector (1.x) or Assertion Consumer Service (2.0) or Security Token Consumer Service (WS-Fed)
Web Agent + Web Agent Option Pack
Policy Server
SAML auth. scheme or WS-Fed auth. scheme

SiteMinder is deployed at both sites. The interactions between the user and each site is similar, where partsco.com is acting as the relying party.

The following illustration is similar for SAML 1.x, SAML 2.0, and WS-Federation; however, the Federation Web Services components are different as follows:

- For SAML 1.x, the Artifact Resolution Service (artifact profile only) is at the IdP and the SAML credential collector is at the SP.
- For SAML 2.0, the Assertion Retrieval Service (artifact binding only) is at the IdP and the Assertion Consumer Service at the SP.
- For WS-Federation, the Single Sign-on Service is at the IdP and the Security Token Consumer Service is at the SP.

  Note: WS-Federation only supports HTTP-POST binding.

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The configuration is similar to Solution 1: Single Sign-on based on Account Linking, except for the following items:

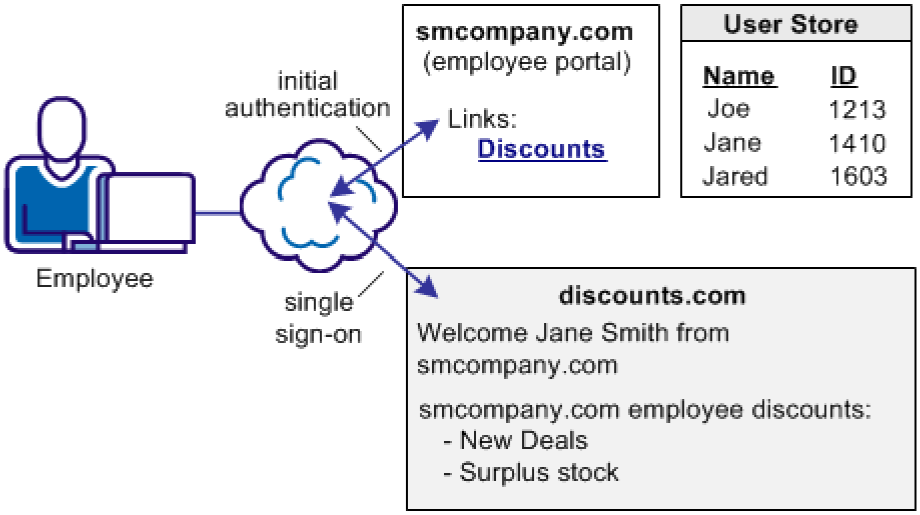
- The administrator at smcompany.com defines the consumer/SP for partsco.com with an attribute specifying the department of the user at the company. The assertion generator includes this attribute as part of the user profile in the assertion it creates for partsco.com.
- The administrator at partsco.com defines an authentication scheme (artifact, post, or WS-federation) for smcompany.com. The scheme extracts the department attribute from the SAML assertion. The scheme then searches the user directory at partsco.com for the user record that matches the department value from the assertion. The administrator defines one user profile record for each department that is allowed to access the partsco.com website.

Use Case 3: Single Sign-on with No Local User Account

In Use Case 3, smcompany.com offers employee discounts by establishing a partnership with discounts.com.

An employee of smcompany.com authenticates at smcompany.com and clicks a link to access discounts.com. The employee is taken to the discounts.com website and presented with the discounts available for smcompany.com employees, without logging in to the discounts.com website.

The following illustration shows this use case.



Discounts.com does not maintain any identities for smcompany.com. The company allows all employees of smcompany.com to access discounts .com as long as long as they have been authenticated at smcompany.com. When an employee of smcompany.com accesses discounts.com, authentication information is sent in a secure manner from smcompany.com to discounts.com. This information is used to allow access to discounts.com.

Additional attributes, such as the user name are passed from smcompany.com to discounts.com to personalize the interface for the individual user.

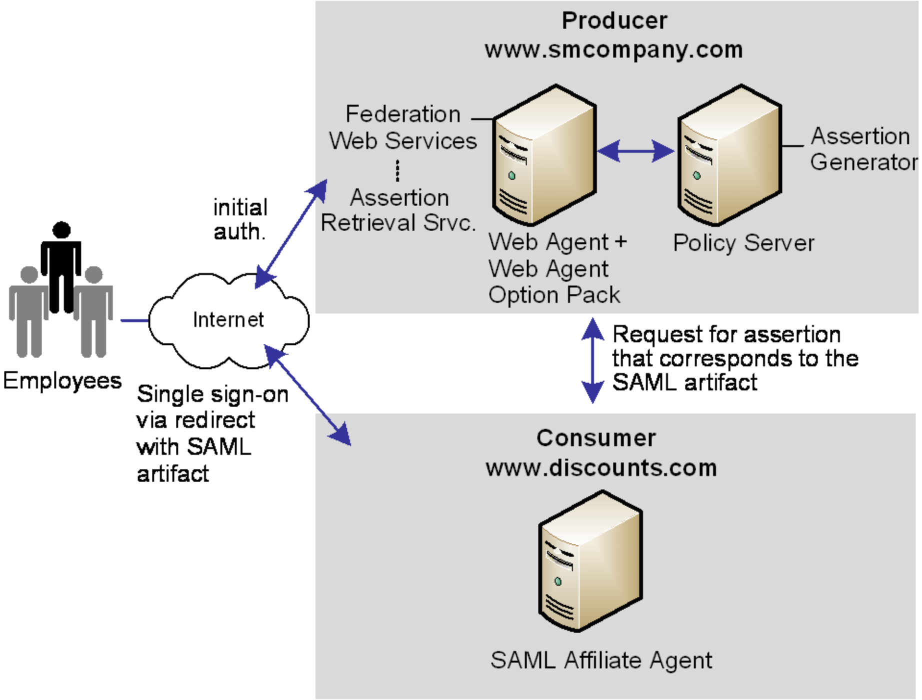
Solution 3: Single Sign-on with no Local User Account

Solution 3 shows how SiteMinder legacy federation can be deployed at smcompany.com and discounts.com to solve Use Case 3: Single Sign-on with No Local User Account (see page 43).

SiteMinder is deployed at smcompany.com by installing the Web Agent with the Web Agent Option pack on one system, and installing the Policy Server on another system. The SAML Affiliate Agent is installed at discounts.com.

Note: The SAML Affiliate Agent only supports SAML 1.0 and is not FIPS-compatible.

The following figure shows single sign-on with no local user account.


Producer
www.smcompany.com
Federation Web Services
Assertion Retrieval Srvc.
Assertion Generator
Web Agent + Web Agent Option Pack
Policy Server
initial auth.
Internet
Employees
Single sign-on via redirect with SAML artifact
Request for assertion that corresponds to the SAML artifact
Consumer
www.discounts.com
SAML Affiliate Agent

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

Smcompany.com is acting as a SAML 1.x producer. When an employee of smcompany.com accesses an employee portal at www.smcompany.com, the following process occurs:

1. The Web Agent provides the initial authentication.
2. When the employee clicks a link at www.smcompany.com to access deals at discounts.com, the link makes a request to the Web Agent at www.smcompany.com.
3. The Web Agent at www.smcompany.com calls the assertion generator. The assertion generator creates a SAML assertion and stores the assertion in the SiteMinder session store. Finally, smcompany.com returns a SAML artifact to discounts.com.
4. The Web Agent redirects the user to www.discounts.com with the SAML artifact in accordance with the SAML browser artifact protocol.

Discounts.com is acting as the consumer site. The SAML Affiliate Agent at www.discounts.com handles the redirect request with the SAML artifact, as follows:

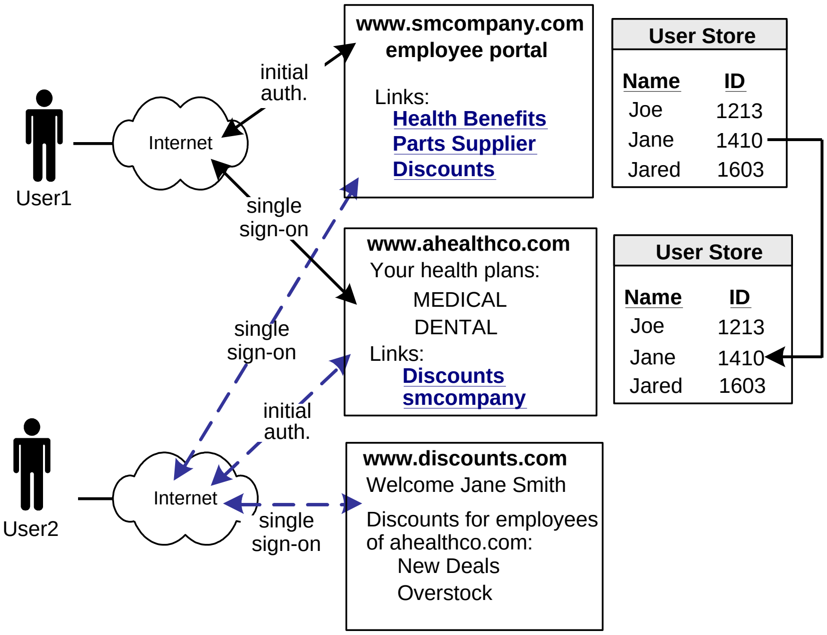
1. The SAML Affiliate Agent obtains the location of the assertion retrieval service at www.smcompany.com from a configuration file.
2. The SAML Affiliate Agent calls the assertion retrieval service at www.smcompany.com.
3. The assertion retrieval service at www.smcompany.com retrieves the assertion from the SiteMinder session store and returns it to the SAML affiliate agent at www.discounts.com.
4. The SAML Affiliate Agent then validates the SAML assertion and issues a SiteMinder affiliate session cookie to the browser.
5. The user is allowed access to resources at discounts.com.

The administrator at smcompany.com uses the Policy Server User Interface to configure an affiliate for discounts.com. The affiliate is configured to include attributes in the assertion. The assertion generator includes the attributes in the SAML assertion it creates for discounts.com.

The administrator at discounts.com configures the SAML Affiliate Agent with information about the discounts.com site, the location of the assertion retriever service at smcompany.com, and the resources the affiliate protects.

Use Case 4: Extended Networks

In Use Case 4, smcompany.com, ahealthco.com, and discounts.com all participate in an extended federated network. This case is an extension of the previous use cases.



In this network, not all customers of ahealthco.com work at smcompany.com. Ahealthco.com provides discounts only to its customers by establishing a relationship between themselves and discounts.com. Ahealthco.com maintains user identities for every customer so ahealthco.com manages local credentials, such as a password for each user. By managing local credentials, ahealthco.com can authenticate users and can provide single sign-on access to its partners.

In this extended network, the users access each website differently:

- User1 accesses health benefit information at the ahealthco.com website. User1 can access the partsco.com website by clicking the PartsSupplier link at smcompany.com, the employee portal. User1 can also click a link at the employee portal to access discounts at discounts.com.

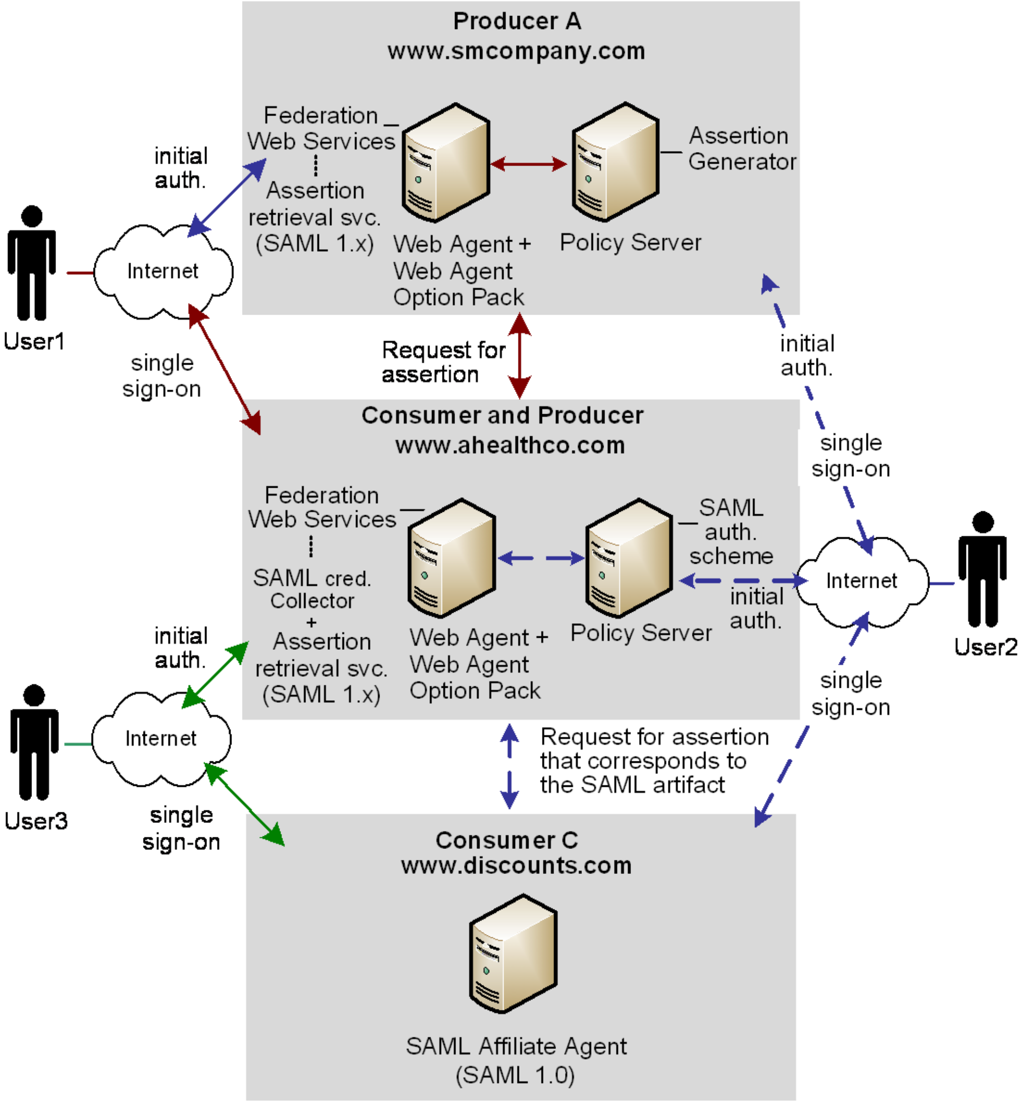
User2 authenticates at the ahealthco.com website and clicks a link to access discounts at discounts.com, without logging in to the discounts.com website. The discounts the site presents to User2 reflect the business arrangement between ahealthco.com and discounts.com. Being employee of smcompany.com, User2 can also click a link at ahealthco.com and access the employee portal at smcompany.com without logging in to website.

- User3 (not shown in the example), is a customer of ahealthco.com, but is not an employee of smcompany.com. User3 authenticates at the ahealthco.com website and clicks a link to access discounts at discounts.com. User3 does not log in to the website. The discounts the site presents to User3 reflect the business arrangement between ahealthco.com and discounts.com. Because User3 is not an employee of smcompany.com, User3 cannot access the smcompany.com website.

Solution 4: Extended Networks

Solution 4 illustrates how legacy federation can be deployed at smcompany.com, ahealthco.com, and discounts.com to solve Use Case 4: Extended Networks (see page 46).

The following illustration shows an extended network. SAML 1.x is the protocol in use.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

SiteMinder is deployed at smcompany.com and ahealthco.com. At each site, the Web Agent and Web Agent Option Pack is installed on one system and the Policy Server on another system. The SAML Affiliate Agent is installed at discounts.com.

In Solution 4:

- smcompany.com acts as a producer for User1 and a consumer for User2.
- ahealthco.com acts as a consumer for User1 and a producer for User2 and a producer for User3.
- discounts.com acts as a consumer for User1, User2, and User3.

The administrator for smcompany.com has configured two entities in an affiliate domain, which represents ahealthco.com and discounts.com. These sites are configured in a similar manner as in Examples 1 and 3 described previously, but the configurations have been extended as follows:

- At smcompany.com, the administrator has configured a SAML authentication scheme (artifact or POST). For User2, the authentication scheme enables smcompany.com to act as a consumer for ahealthco.com.
- At ahealthco.com:
  - The administrator has configured an affiliate object that represents smcompany.com so an assertion is produced for User2. This configuration makes single sign-on to smcompany.com possible.
  - The administrator has configured an affiliate object that represents discounts.com so an assertion is produced for User2 and User3. This configuration makes single sign-on to discounts.com possible.
- At discounts.com, the administrator has configured the SAML Affiliate Agent to act as a consumer for smcompany.com, as in Example 3. An arrow connecting the two sites is not shown in the illustration. The administrator at discounts.com has also added configuration information about ahealthco.com so that the SAML Affiliate Agent can consume assertions from ahealthco.com for User2 and User3.

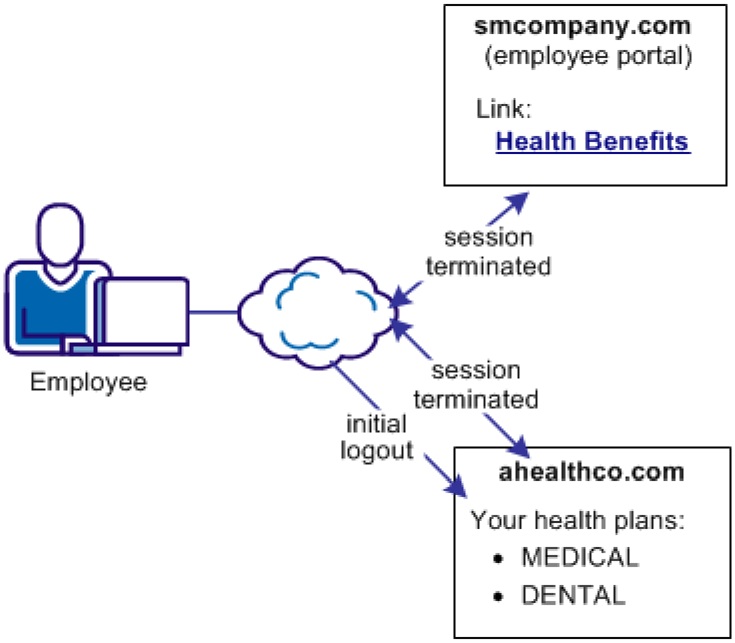
Use Case 5: Single Logout

In Use Case 5, an employee of smcompany.com authenticates at an employee portal, smcompany.com, and selects a link to view the health benefits at ahealthco.com. The employee is taken to the ahealthco.com website and presented with the health benefit information without logging in to the site.

After the employee logs out from ahealthco.com, the site wants to verify the termination of the user session at ahealthco.com and at smcompany.com. Terminating both sessions prevents an unauthorized employee from using the existing session to access resources at smcompany.com or to view benefits of the authorized employee.

Note: The initial logout can occur at smcompany.com and result in both sessions being terminated.

The following illustration shows the use case.



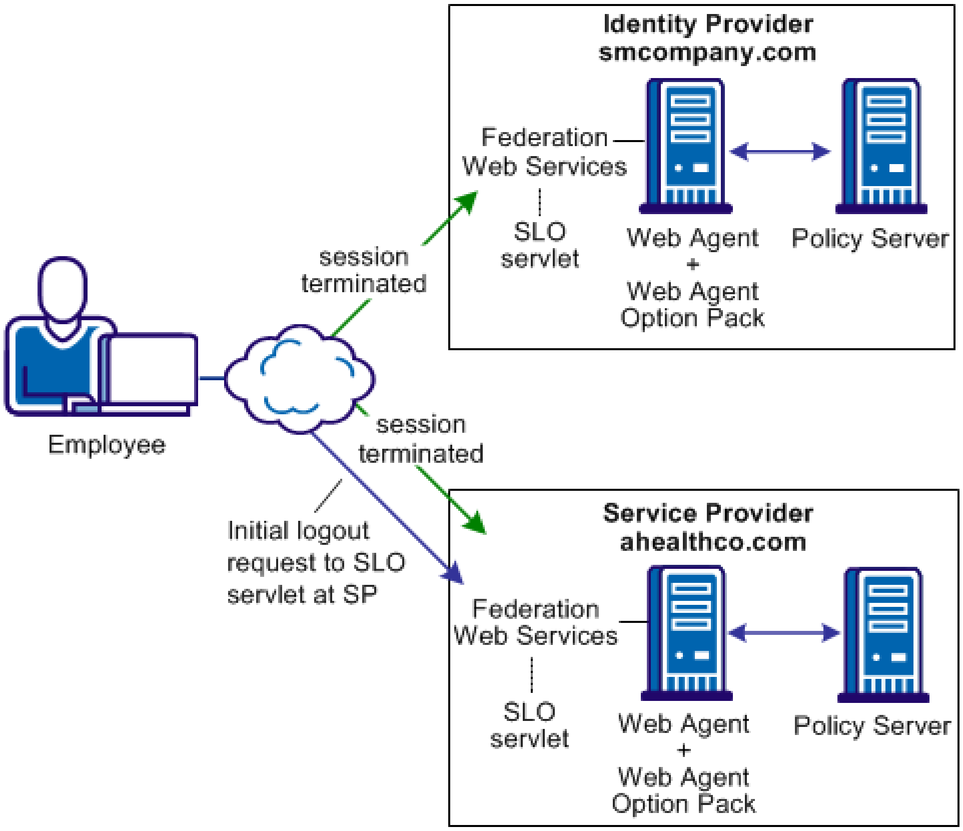
Solution 5: Single Logout (SAML 2.0)

Solution 5 illustrates how federation can be employed to solve Use Case 5: Single Logout (see page 49).

In this solution:

- smcompany.com is the Identity Provider
- ahealthco.com is the Service Provider that initiates the logout request.
- Single logout is enabled at the Identity Provider and the Service Provider.

The following figure shows the SiteMinder solution for single logout.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. Employee performs single sign-on between smcompany.com and ahealthco.com. Smcompany.com places information about ahealthco.com in its session store. Ahealthco.com places information about smcompany.com in its session store.
2. After the employee has finished reviewing health benefits, the employee clicks a log out link at the Service Provider. The browser accesses the single logout servlet at the Service Provider.
3. The user session is terminated from the session store at the Service Provider.

   Note: The termination does not remove the session from the session store; it sets the state to LogoutInProgress.
4. Based on information in the session store, the session is identified as one that an assertion from the Identity Provider, smcompany.com creates.

5. The browser is forwarded to the single logout servlet at smcompany.com, the Identity Provider, with the logout request message as a query parameter.
6. The Identity Provider invalidates the user session from all Service Providers that are associated with that user session, other than ahealthco.com, who initiated the logout request. After all Service Providers confirm the logout, the Identity Provider removes the user session from its session store.

   Note: Other Service Providers are not identified in the illustration.
7. The Identity Provider returns a logout response message to ahealthco.com, the initiating Service Provider, and the user session is removed from the session store.
8. The user is finally sent to a logout confirmation page at ahealthco.com.

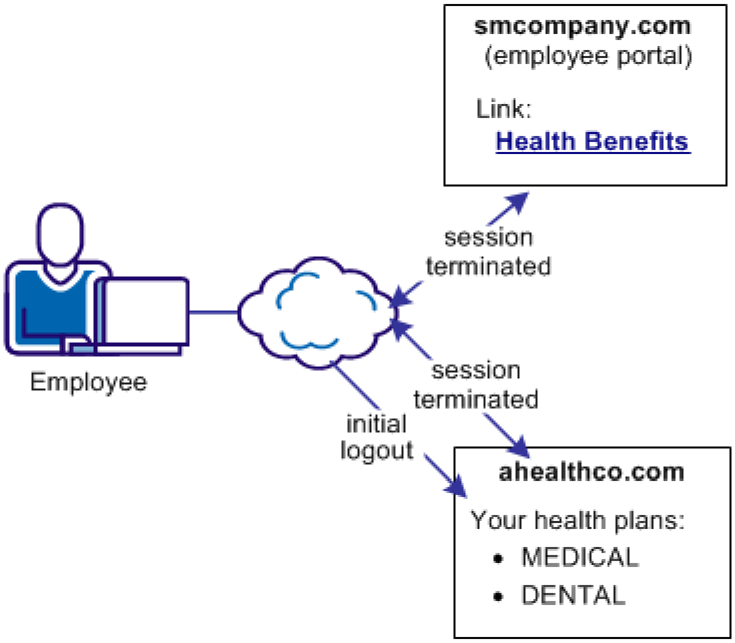
Terminating both sessions prevents an unauthorized employee from using the existing session to view benefits of the authorized employee.

Use Case 6: WS-Federation Signout

In Use Case 6, an employee of smcompany.com authenticates at the employee portal. The employee then selects a link to view health benefits at ahealthco.com. The employee is taken to the ahealthco.com website and presented with the health benefit information without having to sign on to the site.

When the employee logs out from ahealthco.com, ahealthco.com wants the user session at ahealthco.com and the session at smcompany.com terminated. Terminating both sessions helps ensure that an unauthorized employee cannot use the existing sessions to access resources at smcompany.com or to view benefits of the authorized employee.

The following illustration shows the use case.



Solution 6: WS-Federation Signout

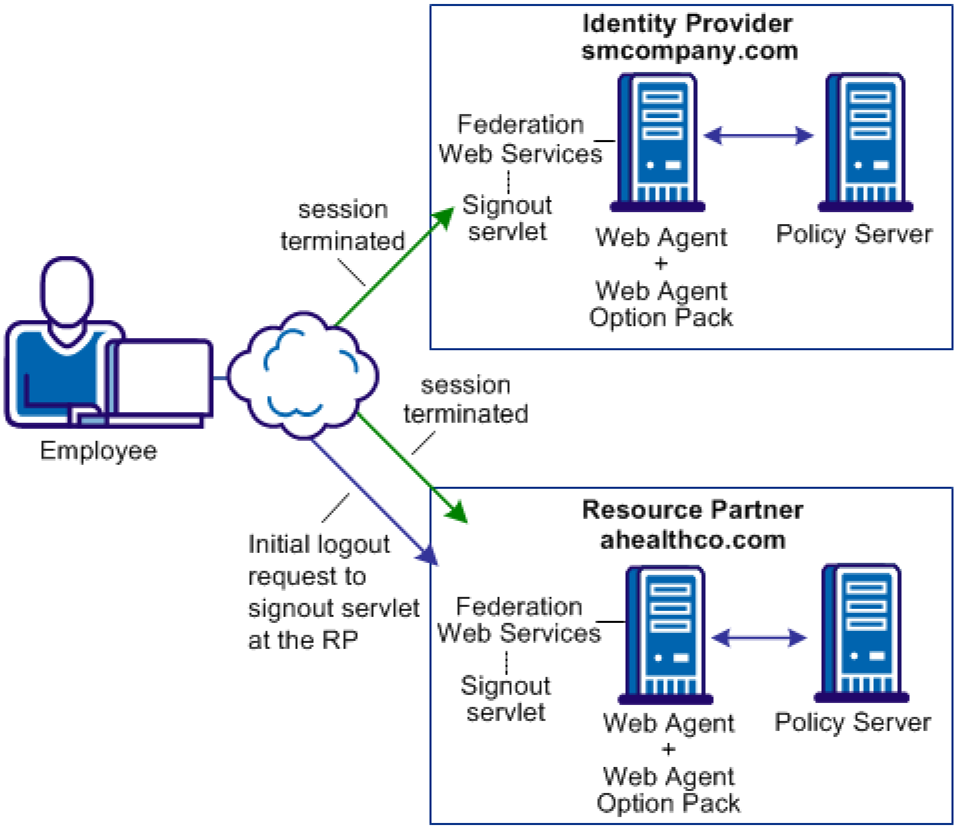
Solution 6 illustrates how federation solves Use Case 6: WS-Federation Signout (see page 52).

In this solution:

- smcompany.com is the Account Partner
- ahealthco.com is the Resource Partner that initiates the signout request.

WS-Federation signout is enabled at the Account Partner and the Resource Partner.

The following figure illustrates WS-Federation sign-out.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. An employee authenticates at smcompany.com and then access their health benefits at ahealthco.com without having to log in. Federated single sign-on is configured between smcompany.com and ahealthco.com. During the transaction, smcompany.com places information about ahealthco.com in its session store. Ahealthco.com places information about smcompany.com in its session store.

2. The employee the health benefit information and clicks a logout link at ahealthco.com. The link calls the signout servlet at smcompany.com.

3. The session of the user is terminated from the session store of the Account Partner. All references to Resource Partners for that user are also removed from the session store.

4. The Account Provider retrieves a SignoutConfirm JSP page, which includes a Signout Cleanup URLs for each Resource Partner.

   The SignoutConfirm page generates a frame-based HTML page with each frame containing a signoutcleanup URL for each Resource Partner that is associated with the user session.

5. The browser of the user then accesses the signout Cleanup URL at ahealthco.com and the session of the user is removed from the session store.

6. The browser of the user is finally sent back to the Account Partner.

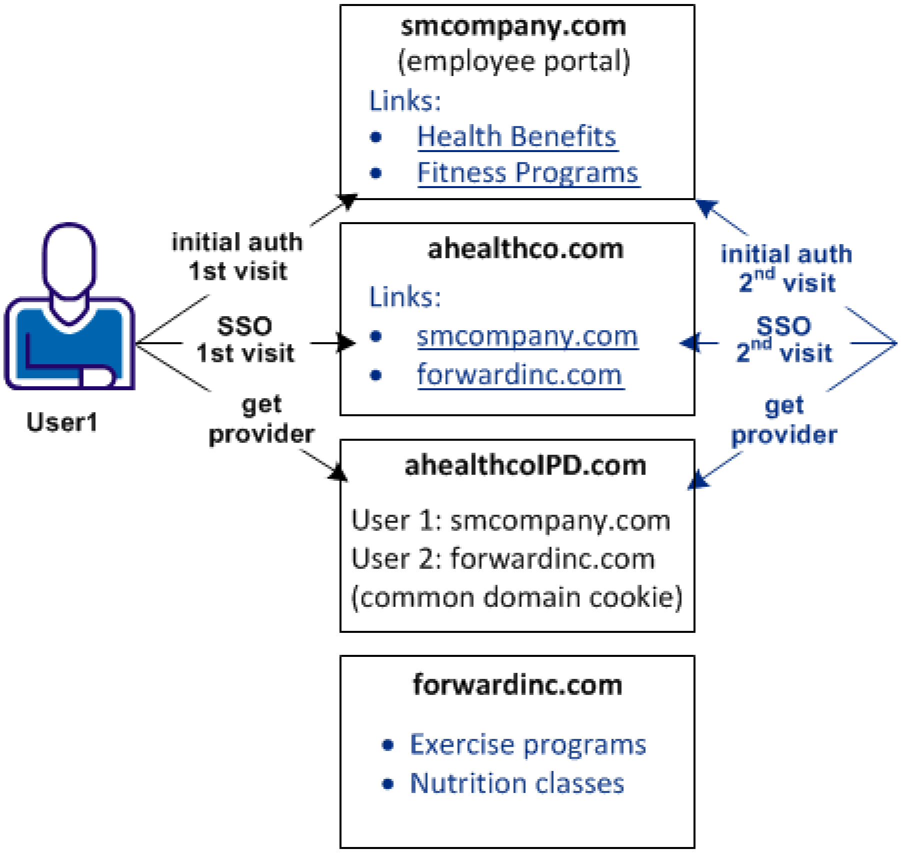
Steps 4-6 are repeated for each Resource Partner simultaneously for complete signout for that user session.

Use Case 7: Identity Provider Discovery Profile

In Use Case 7, several companies contract health benefits from ahealthco.com. When a user requests logs on to ahealthco.com to view their health benefits, ahealthco.com must determine which Identity Provider it sends an authentication request to for a particular user.

IdP discovery is useful in federated networks that have more than one partner providing assertions. It provides a dynamic way for a Service Provider to determine which Identity Provider it sends authentication requests for a particular user.

The following illustration shows a network with the Identity Provider Discovery profile in use.



A user arrives at ahealthco.com. This health provider determines where to send the authentication request. For User1, smcompany.com is where this user authenticates, so this company is set in the common domain cookie at ahealthco.com. For another user, forwardinc.com is an Identity Provider where a user authenticates. Forwardinc.com is set in the common domain cookie at ahealthco.com also.

A prior business agreement between the sites in this network exists so that all sites interact with the Identity Provider Discovery service.

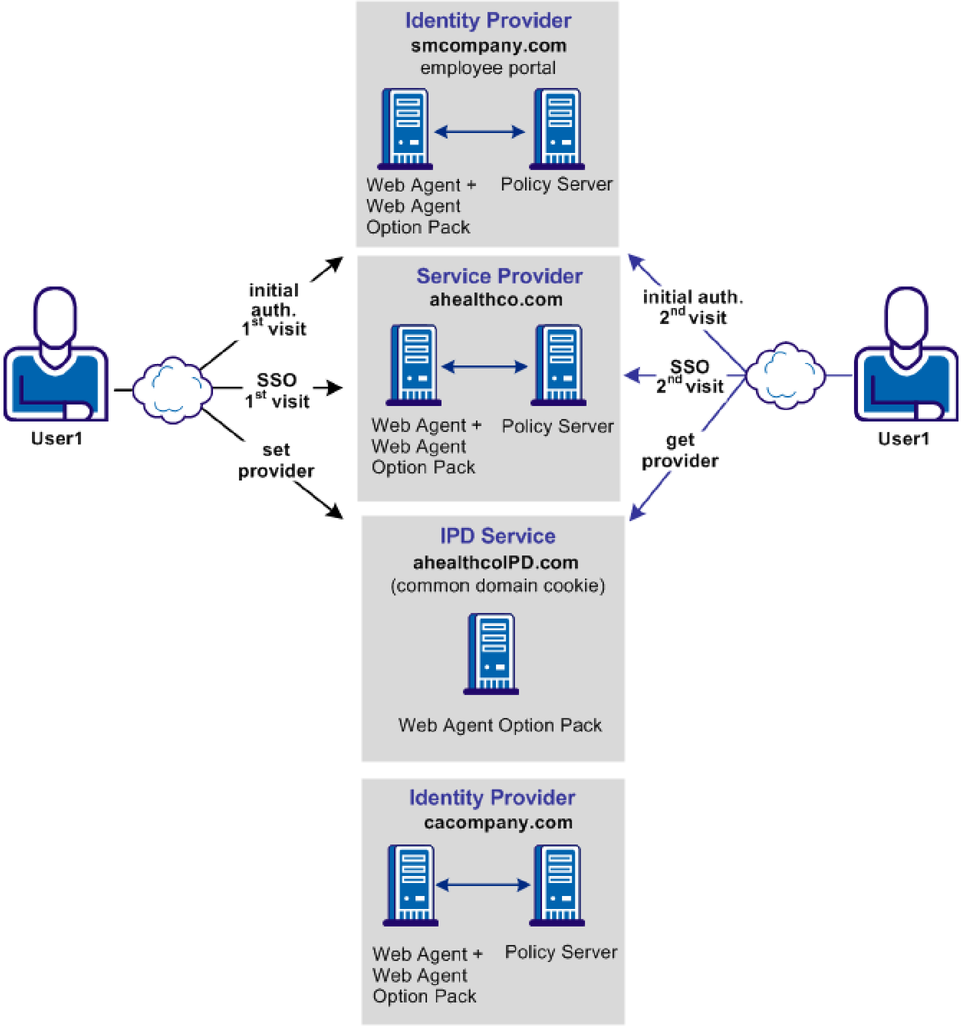
Solution 7: Identity Provider Discovery Profile (SAML 2.0)

Solution 7 illustrates how SiteMinder legacy federation can solve Use Case 7: Identity Provider Discovery Profile (see page 55).

In this solution:

- smcompany.com issues assertions for User 1 and has ahealthco.com configured as its Service Provider.
- ahealthco.com is the Service Provider for smcompany.com and cacompany.com. This site has a SAML 2.0 authentication scheme that is configured for each of these Identity Providers. This site enables single sign-on.
- ahealthcoIPD.com is the Identity Provider Discovery Service for ahealthco.com. The Federation Web Services application, which is installed with the Web Agent Option Pack, provides the IPD service which can read the common domain cookie. This common domain cookie includes all relevant Identity Providers for ahealthco.com.
- cacompany.com is another Identity Provider where users other than User1 can log in.

The following illustration shows the federated network for this solution.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. User 1 initially authenticates at smcompany.com and then logs in to ahealthco.com without having to reauthenticate.

   An agreement exists between smcompany.com and ahealthcoIPD.com to use ahealthcoIPD.com as the IPD service. During the initial authentication process, the Identity Provider URL of smcompany.com is written to the common domain cookie at the IPD service.

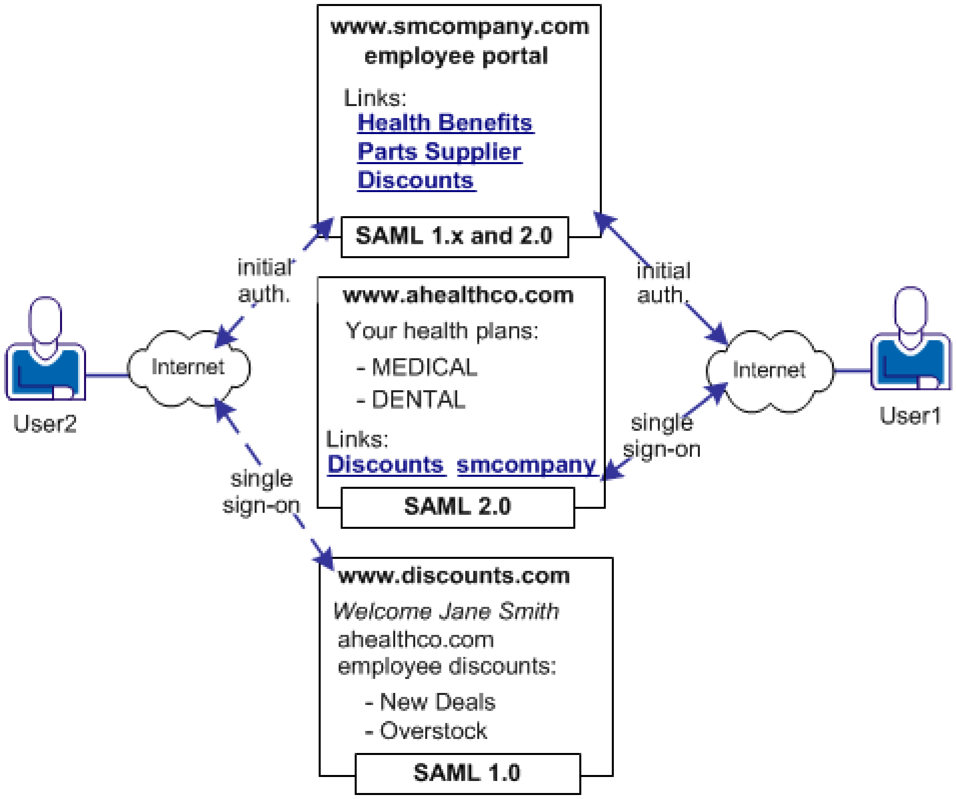
2. User 1, now successfully logged on to ahealthco.com, can look at the health benefits.

3. User 1 then comes to a site selection page at ahealthco.com. A common domain cookie is set for smcompany.com and ahealthco.com is configured to use the IPD service. As a result, ahealthco.com knows that the user previously logged in to smcompany.com. Therefore, ahealthco.com makes the appropriate links available to the user so that user can go back to smcompany.com to log in.

Use Case 8: Multi-protocol Support

In Use Case 8, smcompany.com issues assertions for ahealthco.com and discounts.com. Ahealthco.com uses SAML 2.0 for User1 to communicate between smcompany.com and ahealthco.com. Discounts.com uses SAML 1.0 for User2 to communicate between smcompany.com and discounts.com. The assertions must be suitable for the protocol that the SP uses to consume the assertion.

The following illustration shows the multiprotocol use case.



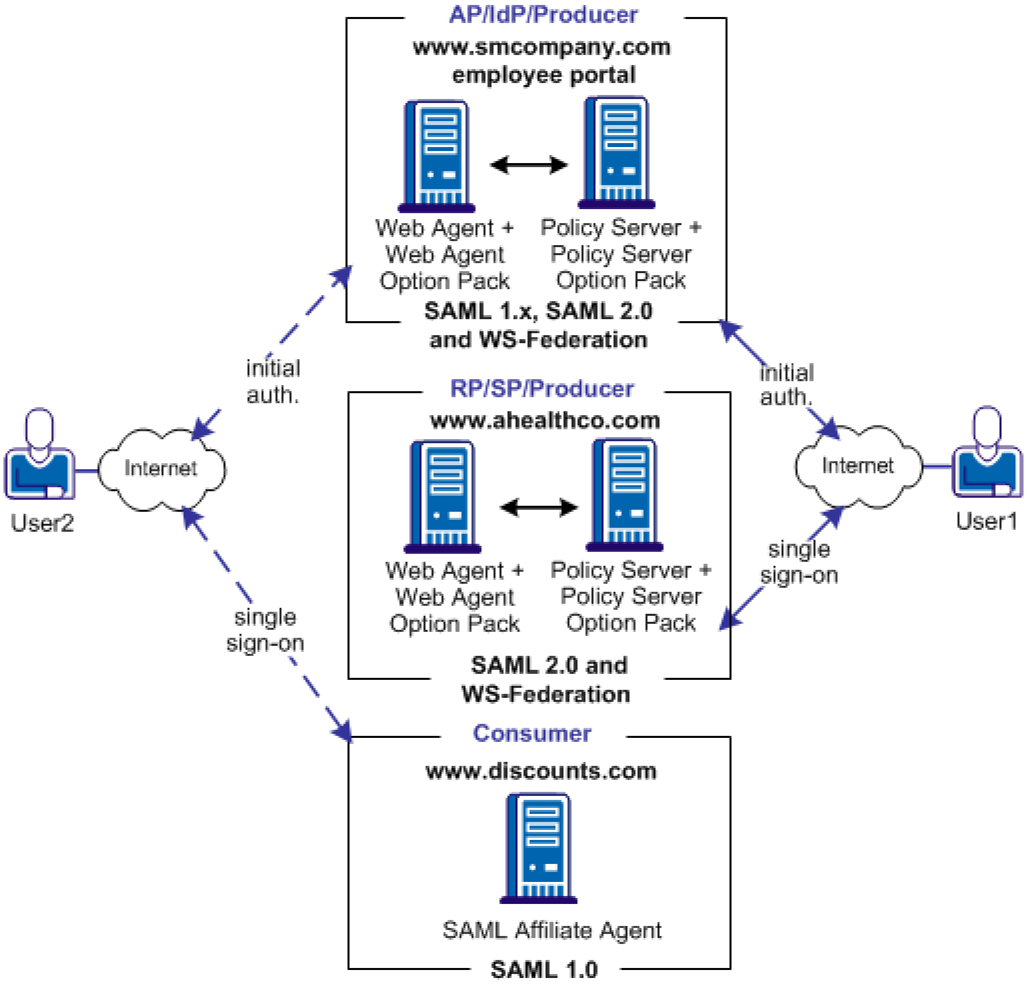
Solution 8: Multi-protocol Network

Solution 8 illustrates how SiteMinder legacy federation can solve Use Case 8: Multi-protocol Support (see page 59).

In this solution:

- For User 1,
  - smcompany.com is the SAML 2.0 Identity Provider for ahealthco.com. The partner, ahealthco.com, is included in an affiliate domain as a SAML 2.0 Service Provider.
  - ahealthco.com is where the SAML 2.0 authentication scheme is configured, and where smcompany.com is identified as the Identity Provider.
- For User 2,
  - smcompany.com is the SAML 1.0 producer for discounts.com, which is a SAML 1.0 consumer. This site uses the SAML Affiliate Agent, which can only consume SAML 1.0 assertions. This site cannot perform any authentication tasks.

The following illustration shows a SiteMinder federated network that implements multiprotocol support.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

In this multiprotocol solution, smcompany.com can issue a SAML 2.0 assertion for User 1 to access resources at ahealthco.com. Additionally, smcompany.com can issue a SAML 1.0 assertion for User 2 to authenticate at discounts.com. Smcompany.com issues an assertion that is based on the session cookie that is set during initial authentication and determines the appropriate protocol for the assertion.

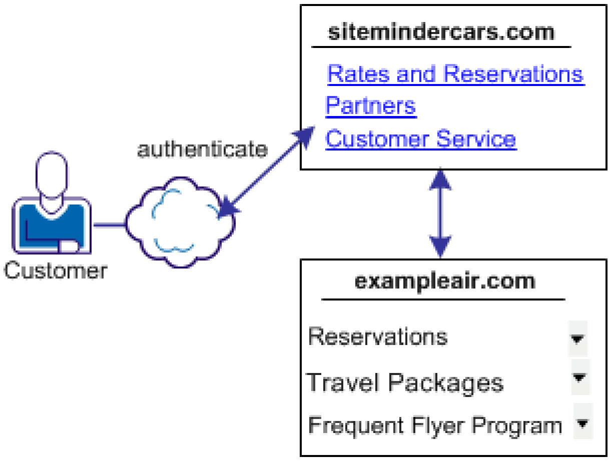
Configure the SAML Affiliate Agent at discounts.com. The smcompany.com is added to its producer information settings in its AffiliateConfig.xml configuration file so that it accepts SAML 1.0 assertions from this site.

Use Case 9: SAML 2.0 User Authorization Based on a User Attribute

In Use Case 9, sitemindercars.com is a car rental service.

A customer of sitemindercars.com logs in and authenticates at sitemindercars.com, then clicks a link to get a quote for a car rental. The customer has a customer profile at this site that includes the frequent flyer number of the customer with exampleair.com. The frequent flyer number of the customer miles determine a certain status level at sitemindercars.com, which offers the customer discounts on car rentals.

The following illustration shows this use case.

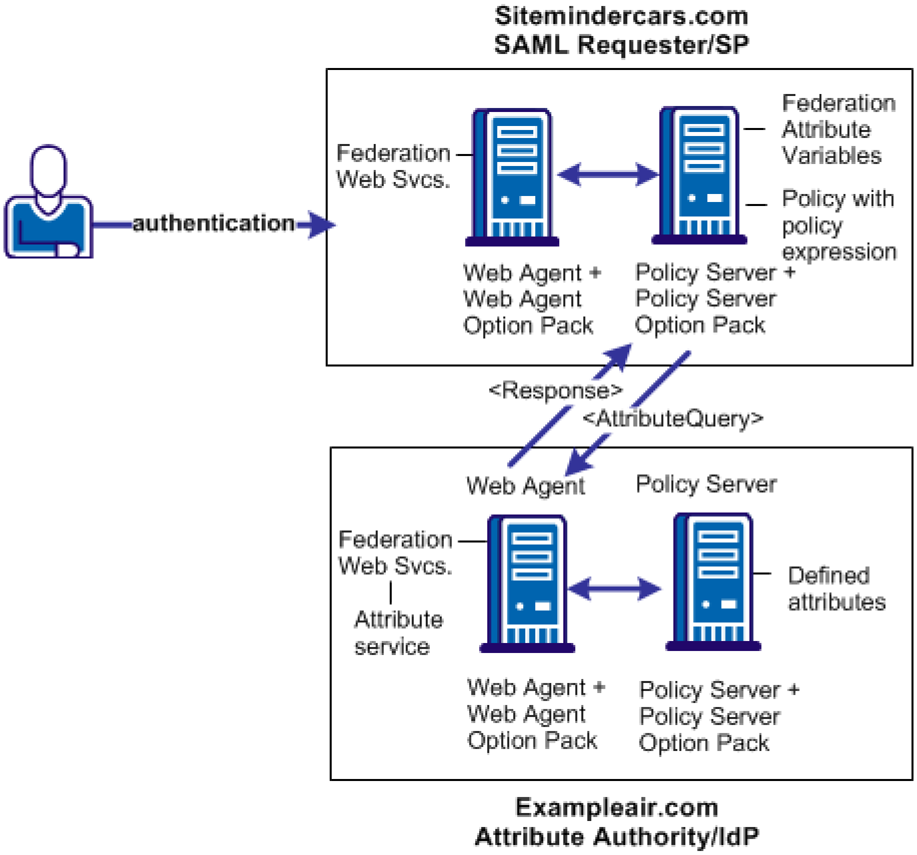


SiteMindercars.com wants to authorize its customers and present the appropriate discount information. The discount information is based on the frequent flyer number of the customer. The company does not want customers having to log in and authenticate at exampleair.com.

Solution 9: SAML 2.0 User Authorization Based on a User Attribute

Solution 9 shows how SiteMinder legacy federation can be deployed at sitemindercars.com and exampleair.com to solve Use Case 9: SAML 2.0 User Authorization Based on a User Attribute (see page 62).

Note: This solution is only for SAML 2.0.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

SiteMinder is deployed at both sites. The Web Agent with the Web Agent Option Pack is installed on one system and the Policy Server with legacy federation on another system. The installations are the same for both sites.

Sitemindercars.com is a Service Provider acting as a SAML Requester. When a customer logs in at this site, the sequence of events is as follows:

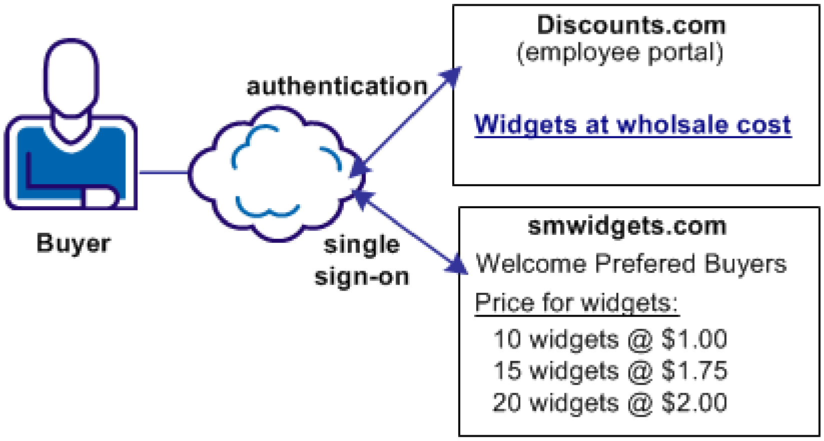
1. The user logs in to sitemindercars.com and the Web Agent provides the authentication.
2. When the user clicks a link to rent a car, the Web Agent first makes a request to the Policy Server at the local site.
3. The Policy Server evaluates the policy expression and identifies unresolved federation attribute variables. The Policy Server tries to resolve the variable by looking up the user in the user directory. The Policy Server looks in the directory associated with the policy protecting the requested URL.
4. The local Policy Server cannot resolve the user attribute variable. The Policy Server locates the NameID configuration for the Attribute Authority, Exampleair.com.
5. The Policy Server sends an <AttributeQuery> containing the NameID and the frequent flyer attribute to the Exampleair.com, which is acting as the Attribute Authority.
6. Exampleair.com returns a response to sitemindercars.com that contains an attribute assertion which includes the requested attribute.
7. The SAML Requester resolves the variables and evaluates the policy expression, returning authorization status to the Web Agent.
8. The Web Agent allows access to the appropriate resource.

Use Case 10: SAML 2.0 Single Sign-on with No Name ID at the IdP

In Use Case 10, discounts.com purchases widgets from smwidgets.com.

A buyer for discounts.com clicks on a link to access the latest widget price list at smwidgets.com. The buyer is taken to the smwidgets.com website and presented with the price list without having to log in to the discounts.com website.

The following illustration shows this use case.



Discounts.com requests access to a price list at smwidgets.com. There are no federated buyer user identities stored locally or at smwidgets.com. Discounts.com sends a request to smwidgets.com. When smwidgets.com receives the request, it creates an identity for the buyer. The identity is sent back in a secure manner to discounts.com. Discounts.com uses this identity to authenticate the user and allow the buyer access to the requested resource.

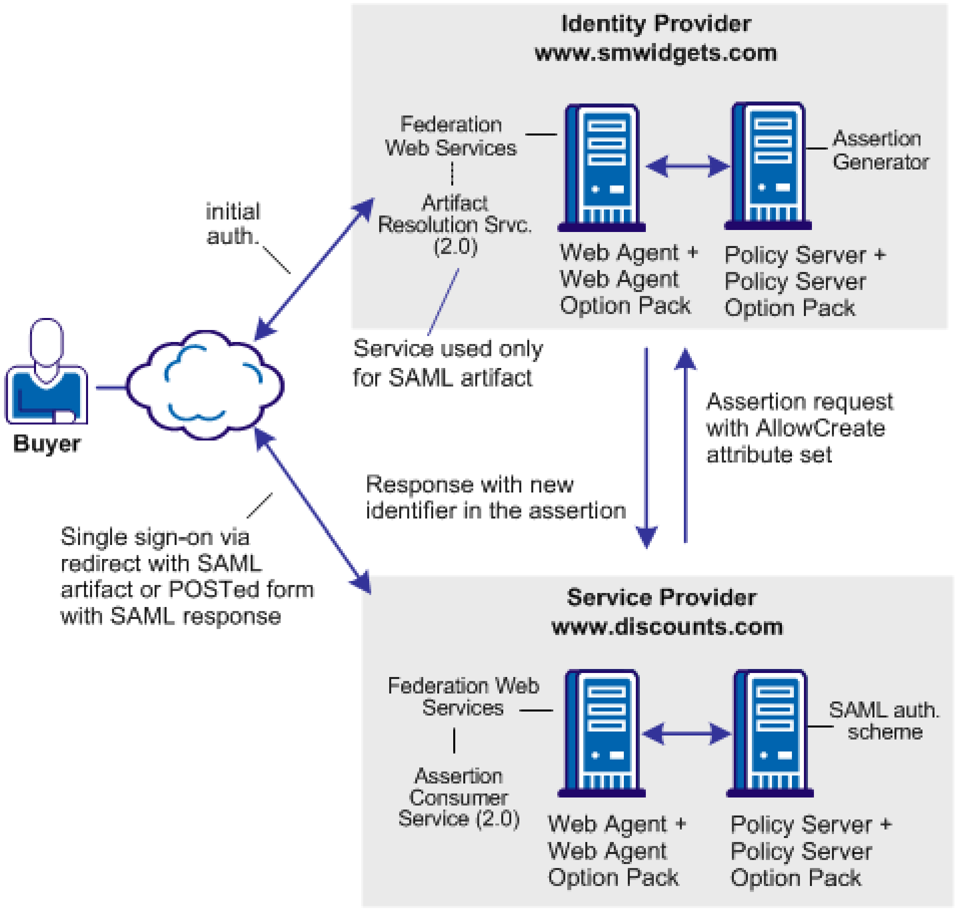
REVIEWERS: The paragraph above sayd that discounts.com sends a request to smwidgets.com. What kind of a request--an authentication request?

Solution 10: Single Sign-on with No Name ID at the IdP

Solution 10 shows how SiteMinder legacy federation can be deployed at smcompany.com and discounts.com to solve Use Case 10: SAML 2.0 Single Sign-on with No Name ID at the IdP (see page 64).

SiteMinder is deployed at discounts.com and smwidgets.com. A Web Agent and Web Agent Option pack is installed on one system, and the Policy Server with legacy federation is installed on another system.

In the following illustration, smwidgets.com is acting as the Identity Provider and discounts.com is acting as the Service Provider.


Identity Provider
www.smwidgets.com
Federation Web Services
Artifact Resolution Srvc. (2.0)
Assertion Generator
Web Agent + Web Agent Option Pack
Policy Server + Policy Server Option Pack
initial auth.
Service used only for SAML artifact
Buyer
Assertion request with AllowCreate attribute set
Response with new identifier in the assertion
Single sign-on via redirect with SAML artifact or POSTed form with SAML response
Service Provider
www.discounts.com
Federation Web Services
Assertion Consumer Service (2.0)
SAML auth. scheme
Web Agent + Web Agent Option Pack
Policy Server + Policy Server Option Pack

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

For single sign-on between the two sites, where there is no federated user identity at the Identity Provider, the sequence of events is as follows:

1. The user clicks on a link at discounts.com to proceed to the target site. This link initiates a call to the local Policy Server to generate an authentication request. In this request, an optional attribute named AllowCreate has been included, based on the configuration of the SAML 2.0 authentication scheme at the Service Provider.
2. The Federation Web Services application at the local Web Agent redirects the request to the Single Sign-on service at the IdP, smwidgets.com.

3. The request is then forwarded to the IdP Policy Server, which generates an assertion. During assertion generation, the Policy Server searches for the attribute associated with the user requesting access. For example, the telephone number attribute can be requested as the value of the Name ID.

   If the Policy Server cannot find a value for the telephone number attribute, it verifies its configuration for the AllowCreate option. If this option is configured, the Policy Server searches the authentication request from the Service Provider to see if the AllowCreate option exists.

   If the Allow/Create feature is enabled at both sites, the Policy Server generates a new identifier for the user attribute. The Policy Server places that identifier in its user store.

   Note: The identifier is the value of the user attribute.

4. The assertion is returned in a response message to the IdP Web Agent (FWS). The IdP returns a form to the browser containing the response, the Assertion Consumer URL, and the javascript to submit the form.

5. The form is posted to the Assertion Consumer Service at the Service Provider. The Service Provider uses the response message to log in to the Policy Server, using the response as credentials.

6. The Service Provider at smwidgets.com validates the credential by looking for the attribute in its user store. Assuming that it finds the user, the user is logged in by the SAML authentication scheme.

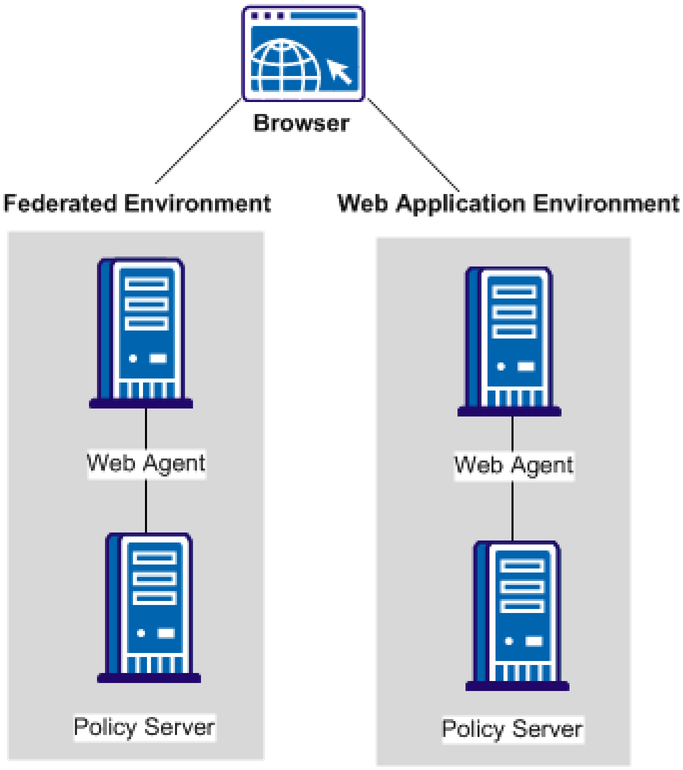
7. The SP Web Agent creates an SMSESSION cookie for the smwidgets.com domain. The Agent places the cookie in the browser and redirects the user to the target destination.

Use Case 11: SAML Artifact SSO Using Security Zones

In use case 11, CompanyA, the producer site, wants to protect Web Agent applications and federated partner resources. The protocols that CompanyA uses for federated single sign-on are the SAML 2.0 artifact profile and SAML 2.0 single logoff.

For federated resources, a persistent user session is required because the SAML artifact profile stores assertions in the session store at the producer-side Policy Server. Consequently, calls are made to the session store to retrieve the assertion, impacting performance.

The following illustration shows a producer site that combines a federated environment and a web application environment.

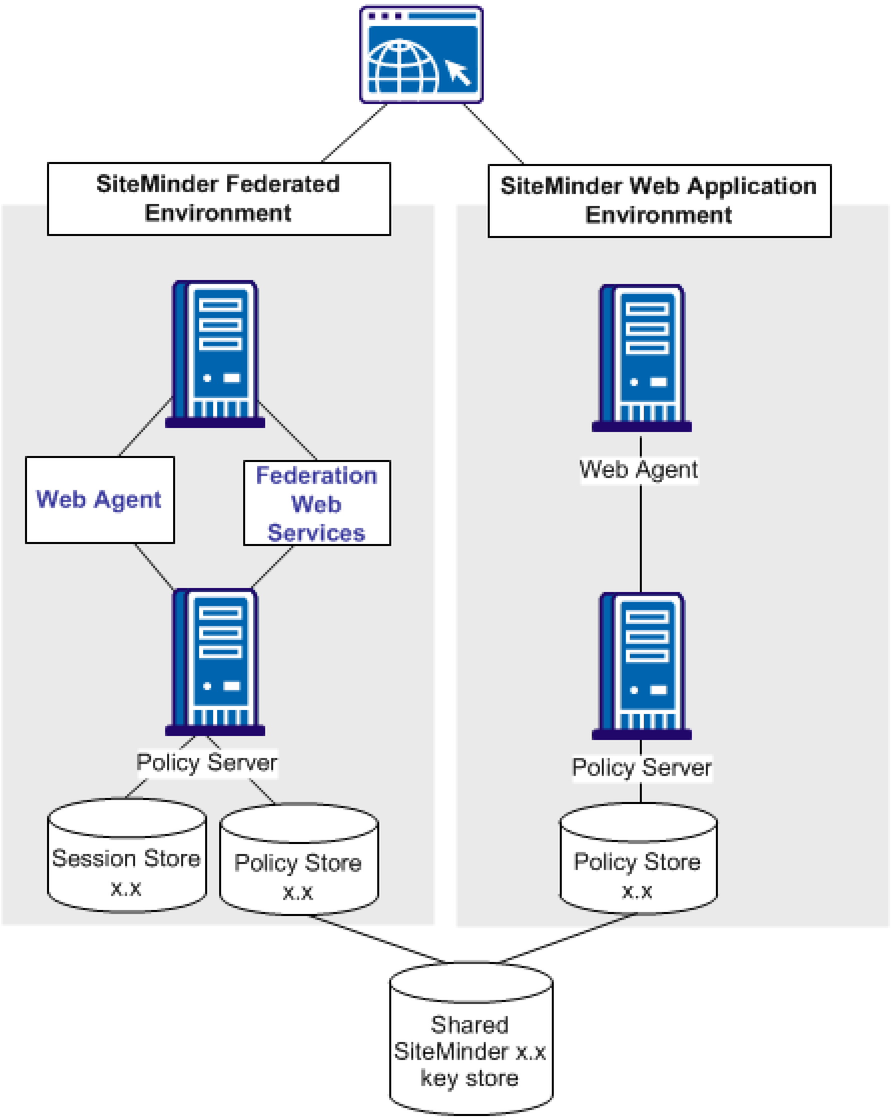


Solution 11: SAML Artifact SSO Using Security Zones

Solution 11 illustrates how you can set up a parallel web application and federation environments to solve Use Case 11.

A security zone is a segment of a single cookie domain, which is used as a method of partitioning applications. You can assign different security requirements to each zone. Producer-side Web Agents that protect requested federated resources enforce security zones. SiteMinder security zones eliminate the need for a persistent user session to be associated with every request for HTTP-Artifact protected applications.

The following figure illustrates a deployment that uses two different SiteMinder environments at a single asserting party. One SiteMinder environment is for federation functionality and the other is for web application protection.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The figure reflects the following set up:

Web application environment

Agent Configuration Object or Local Configuration File	**Trusted Security Zones**	**Cookies Read by the Web Agent for the Zone**
DefaultAgent	SM (default)--primary zone	The DefaultAgent configuration enables the Web Agent to read and write the default session cookie, SMSESSION.

Federation Environment

Agent Configuration Object or Local Configuration File	**Trusted Security Zones**	**Cookies Read by the Web Agent for the Zone**
FedWA used by the Web Agent	FED--primary zone SM--additionally accepted zone	The FedWA configuration enables the Web Agent to read and write SESSIONSIGNOUT cookies, and read SMSESSION cookies.
FedFWS used by the FWS application	FED--primary zone only	Configures the FWS to read and write SESSIONSIGNOUT cookies.

All resources protected in the web application environment use non-persistent user sessions. As a result, when users are authenticated and authorized, the SMSESSION cookie contains a non-persistent user session specification. The non-persistent session specification helps ensure that requests to web applications do not incur the performance penalty of calling the session store.

When the Web Agent in the federation environment receives a request, this request is directed to the Authentication URL to establish a user session. The user making the request already has an SMSESSION cookie from the prior authentication in the web application environment. However, the user has no SESSIONSIGNOUT cookie.

The Web Agent in the federation environment writes a SESSIONSIGNOUT cookie. The SESSIONSIGNOUT cookie has a persistent user session specification and it uses the same session ID as the SMSESSION cookie. This persistent user session protects the Authentication URL, which authenticates users federating to a partner site.

The Web Agent in the federation environment reads the SMSESSION cookie and writes a SESSIONSIGNOUT cookie in accordance with the security zones associated with the FedWA configuration.

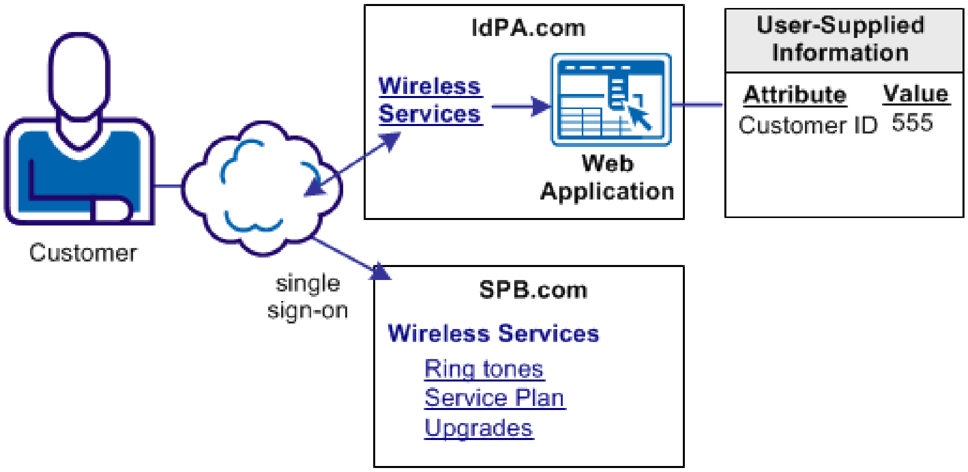
The Federation Web Services application in the federation environment reads the SESSIONSIGNOUT cookie. Because the cookie contains a persistent user session, a call to the session store is not necessary. The Federation Web Services application successfully processes the federation request that requires a persistent user session.

Use Case 12: SAML 2.0 SSO Using Attributes from a Web Application

In use case 12, an Identity Provider, IdPA.com, wants to include web application attributes in an assertion. This use case is only applicable to SAML 2.0 deployments.

For this use case, single sign-on can be initiated at the Identity Provider or the Service Provider. The profiles that IdPA.com uses is SAML 2.0 (POST and Artifact) and WS-Federation.

The following figure shows an example of this use case.



IdP-initiated Single Sign-on with Web Application Attributes

IdPA.com has a web application that allows access to protected resources at its business partner SPB.com. When the customer logs in at IdPA.com, they select a link for the business partner. The user is sent to the web application, where they are prompted to enter a customer ID. IdPA.com must send this information to SPB.com so that the customer is permitted access to the requested resource.

SP-initiated Single Sign-on with Web Application Attributes

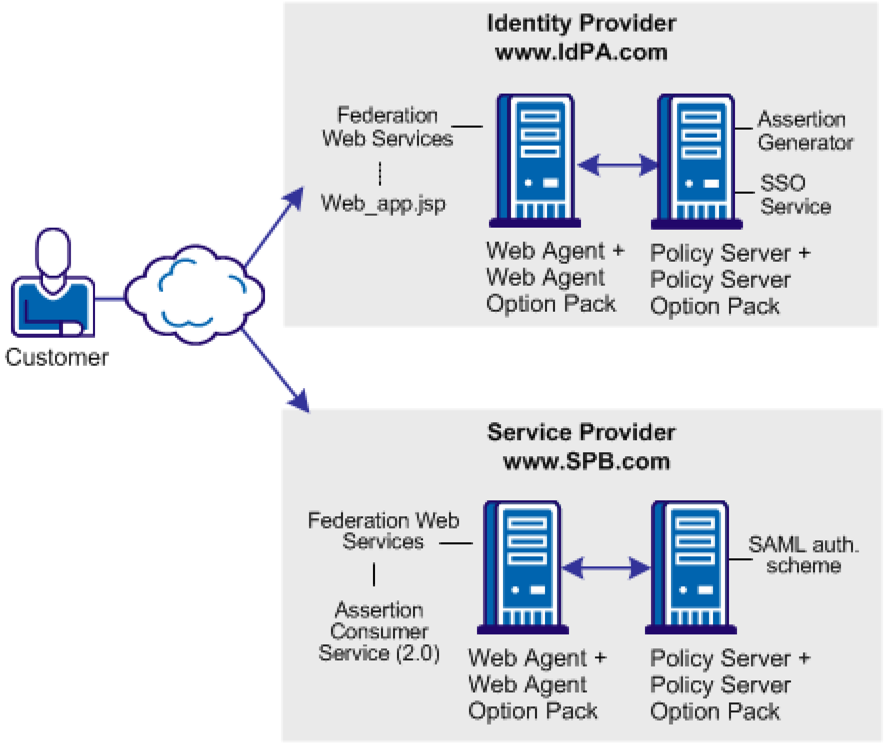
A customer selects a link at SPB.com, the Service Provider. This link is for protected resources, so the customer is redirected to IdpA.com to be authenticated. After the customer successfully authenticates at IdPA.com, the IdP redirects the customer to the web application where the customer provides specific user information. Upon submitting the information, the customer is returned to SPB.com to complete single sign-on for the requested resource.

Solution 12: SSO with Attributes from a Web Application

Solution 12 shows how SiteMinder legacy federation can be deployed at IdPA.com and SPB.com to solve Use Case 12 (see page 71).

SiteMinder is deployed at both sites. At each site, the Web Agent and the Web Agent Option pack are installed on one system, and the Policy Server is installed on another system.

In the following illustration, IdPA.com is the Identity Provider and SPB.com is the Service Provider and single sign-on is initiated at the Identity Provider.


Identity Provider
www.IdPA.com
Federation Web Services
Web_app.jsp
Assertion Generator
SSO Service
Web Agent + Web Agent Option Pack
Policy Server + Policy Server Option Pack
Customer
Service Provider
www.SPB.com
Federation Web Services
Assertion Consumer Service (2.0)
SAML auth. scheme
Web Agent + Web Agent Option Pack
Policy Server + Policy Server Option Pack

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

For IdP-initiated single sign-on, the sequence of events is

1. At the IdP, the user clicks the web page link and *one* of the following actions occurs:

   - The user is directed to Single Sign-on (SSO) service from IdPA.com. This service recognizes that the user does not have a session. The user is redirected to the IdP and is prompted to log in. After successful login, the user is redirected back to the SSO service. An application URL defined for the SSO service instructs the SSO service to send the user to a custom web application.

     Important! When the user goes to the SSO Service first, several query parameters (SPID, ProtocolBinding, RelayState) are included with the original SSO request. The SSO service groups this query data into one query parameter named SMPORTALSTATE, and then redirects the user (through a GET) to the web application.

   - The user logs in and is taken directly to the web application.

   Note: After the user is locally authenticated at the IdP, the user is never redirected to the Authentication URL if the session is valid.

2. The web application prompts the user supplies the requested information. These attributes are posted to the SSO Service.

   Important! The web application must maintain and POST the SMPORTALSTATE query parameter and the collected attributes back to the SSO Service.

3. The SSO service processes the SAML request and unpacks the data from the SMPORTALSTATE parameter. The service takes this data with the attributes from the web application and passes all the POST data to the assertion generator.

4. The Assertion Generator creates the assertion.

   Important! The SSO Service makes all the attributes *available* to the Assertion Generator. Write and configure the assertion generator plug-in to add the attributes to the assertion.

5. After the IdP generates the assertion, the IdP redirects the user to the Assertion Consumer Service at the Service Provider. The Service Provider processes the assertion.

6. The user gains access to the requested resource at the Service Provider.

For SP-initiated single sign-on:

1. At the SP, the user clicks on a link and an AuthnRequest is sent to the Single Sign-on (SSO) service at the Identity Provider.

   Note: In the SP-initiated single sign-on, the request must arrive at the SSO service directly from the SP, as the SAML specification requires. The user cannot go directly to the web application.

2. At the IdP, the SSO service recognizes that the user does not have a session. The user is redirected to the Authentication URL to authenticate and establish a session. After the user establishes a session, the IdP redirects the user back to the SSO service. An application URL defined for the SSO service instructs the SSO service to return the user to a custom web application.

   Important! When the user is directed to the SSO Service, several query parameters (SPID, ProtocolBinding, RelayState) are included with the original request. The SSO service groups this query data into one query parameter named SMPORTALSTATE, and then redirects the user (through a GET) to the web application.

3. The web application prompts the user to supply the requested information. These attributes are posted to the SSO Service.

   Important! The web application must maintain and POST the SMPORTALSTATE query parameter and the collected attributes back to the SSO Service.

4. The SSO service processes the SAML request and unpacks the data from the SMPORTALSTATE parameter. The service takes the data and the attributes from the web application and passes all the POST data to the assertion generator.

5. The IdP generates the assertion and includes all the attributes. The IdP redirects the user to the Assertion Consumer Service at the Service Provider, where the assertion is processed.

   Note: Write and configure an assertion generator plug-in to add the attributes to the assertion.

6. The user gains access to the requested resource at the Service Provider.

Configure SSO with Attributes from a Web Application

Configuring single sign-on based on attributes from a web application, requires specific steps.

Follow these steps:

1. Create a custom web application for the IdP in your network. This custom application can prompt the user for as many attributes as required. Conversely, the application can supply standard attributes and not prompt the user for any information. How attributes are gathered is entirely dependent on how the custom application is written.

   Important! For IdP-initiated single sign-on, if the user is directed to the web application before the SSO service, the web application must include the parameter **AllowApplicationPost=yes**. The SSO service accepts the post as long as the application includes the AllowApplicationPost parameter.

   The SiteMinder Web Agent Option Pack comes with sample JSP applications that you can use as a basis for your custom web application. The path to the sample JSP applications is: *web_agent_home*/affwebservices/. The sample applications are:

   sample_application.jsp

   This sample application can be used for IdP- or SP-initiated single sign-on. The user is first directed to the SSO Service and then sent to the custom web application. This application can be entered for the Application URL in the Service Provider Properties (SAML 2.0) dialog or the Resource Provider Properties (WS-Federation) dialog.

   unsolicited_application.jsp

   This sample application can be used for IdP-initiated single sign-on. The user is sent directly to the web application and not to the SSO Service. The application assumes that the user is already authenticated at the Identity Provider.

   Note: This file shows how to use the AllowApplicationPost parameter in an application.

2. (Optional) If the user is initially directed to the IdP SSO service:

   a. Specify an Application URL in the SAML 2.0 authentication scheme.

   b. Configure the Assertion Generator plug-in to add the attributes to the assertion.

3. (Optional) If the user is sent directly to the custom web application from the IdP, you do not have to provide a value for the Application URL parameter. However, write and configure the assertion generator plug-in to work with SiteMinder. See the *Programming Guide for Java* for information about creating an assertion generator plug-in.

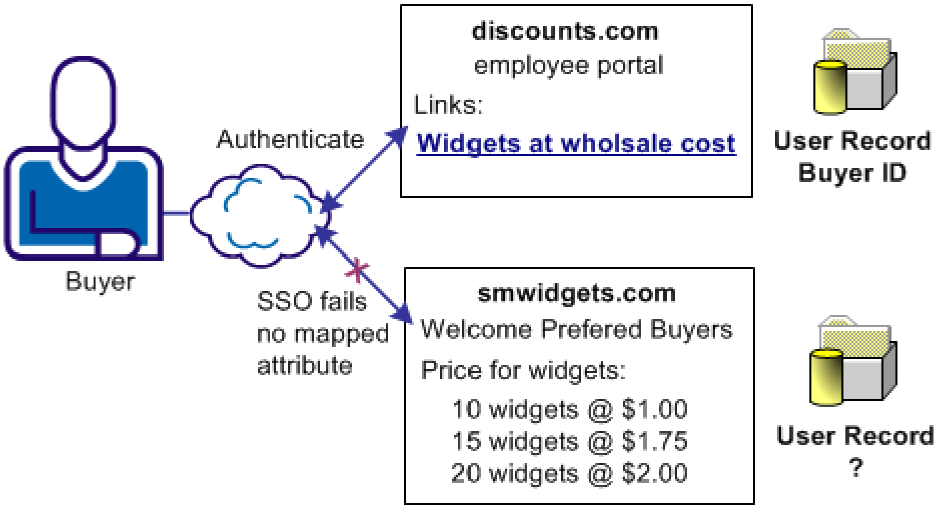
Note: The order of the procedure steps is provided as a guideline. You can perform these steps in a different order.

Use Case 13: SSO with Dynamic Account Linking at the SP

In Use Case 13, the IdP, discounts.com, includes an attribute named buyerID that identifies a particular user and is included in an assertion. When the assertion is sent to the Service Provider, smwidgets.com, the same attribute does not exist in the user record at the Service Provider. The Service Provider must create an attribute in the appropriate user record so that the user can authenticate and gain access to the protected resource.

An employee of discounts.com selects a link to access the latest price list on widgets at smwidgets.com. The employee logs in with the name and buyer ID.

The following illustration shows this use case.



The identity that is based on the buyer ID of the user is created at discounts.com and placed in the assertion. The buyer ID value is entered as the NameID in the assertion. However, there is no mapped identity at smwidgets.com for the buyer ID. The administrator at the Service Provider establishes a mapping. The mapping has to use dynamic account linking so that smwidgets can authenticate the employee and can allow the employee access to the price list.

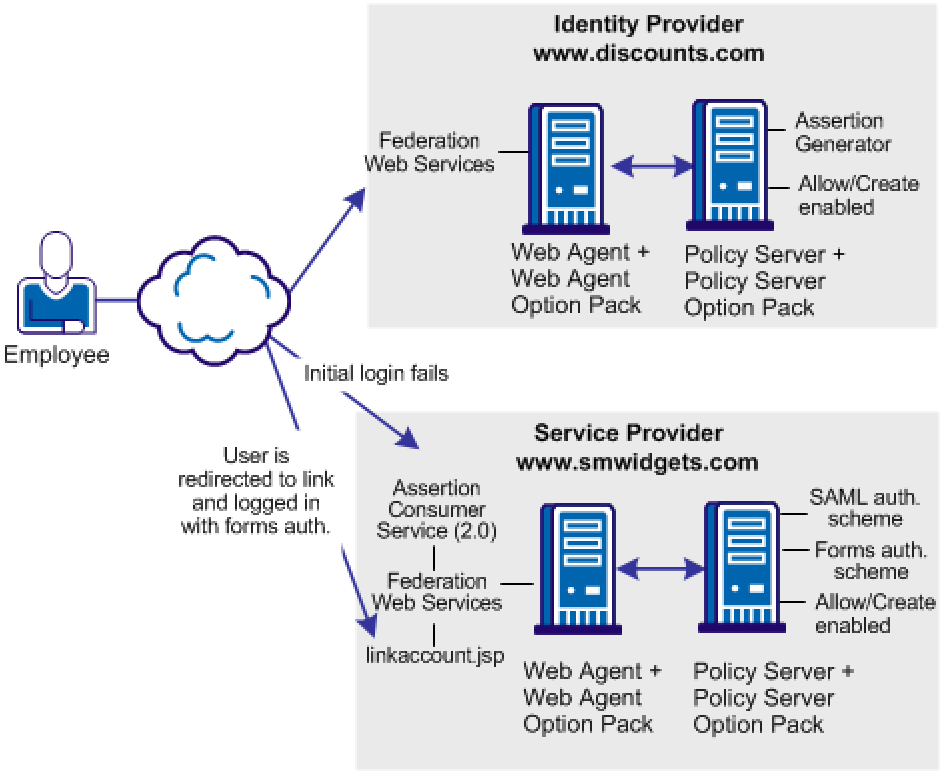
Solution 13: SAML 2.0 SSO with Dynamic Account Linking at the SP

Solution 13 shows how SiteMinder legacy federation can be deployed at IdPA.com and SPB.com to solve Use Case 13 (see page 76).

Note: Dynamic account linking is only supported with SAML 2.0.

SiteMinder is deployed at both sites. Each site has a Web Agent and Web Agent Option Pack installed on one system, and the Policy Server on another system.

The following illustration shows single sign-on with dynamic account linking at the Service Provider.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The order of events for this solution is

1. The employee initially logs in and authenticates at discounts.com. Discounts.com creates an assertion for the employee. Discounts.com posts the assertion (POST binding) or redirects the user with an artifact (artifact binding) to the Assertion Consumer Service at smwidgets.com. This assertion includes an attribute named buyerID.

2. The Assertion Consumer Service at smwidgets.com tries to authenticate the user with the SAML authentication scheme. However, the buyerID attribute of the employee does not map to a local user record so the authentication fails.

3. As part of the SAML authentication scheme at the SP, a redirect URL is defined, which points to the directory *web_agent_home*/affwebservices/linkaccount.jsp. The employee is redirected to this URL.

   Note: The linkaccount.jsp file must be part of a protected realm. The default location for this file is http://*sp_home*/affwebservices/public/. Copy the file from this location to a protected realm.

4. A Web Agent that authenticates the local user with the forms authentication scheme protects this linkaccount.jsp URL. After a successful authentication, a SiteMinder session at smwidgets.com is established and an SMSESSION cookie is placed in the browser of the employee.

5. The linkacount.jsp gets loaded in the browser and the user sees a message to permit the link to the SP account. Click on the button to permit the account linking.

6. The user is redirected to the Assertion Consumer Service, where the browser of the employee presents the SMSESSION cookie with the assertion.

7. The Assertion Consumer Service extracts the NameID from the assertion and inserts the NameID value into a newly created buyerID attribute. The buyerID attribute is in the existing user record of the employee. The Assertion Consumer Service knows which user record to map because the UserDN in the SMSESSION cookie identifies the user.

   The search specification configured in the SAML 2.0 authentication scheme indicates which attribute is mapped to the NameID. In this case, the search specification is buyerID=%s.

8. After the attribute is mapped, the SAML authentication scheme authenticates the user that is based on the assertion and establishes a new user session.

   The next time that the same user presents an assertion with the buyer ID, the user successfully gains access to the requested resource.

Configure SAML 2.0 SSO with Dynamic Account Linking at the SP

Configure several components at the Service Provider to enable SAML 2.0 single sign-on with dynamic account linking:

- AllowCreate feature

  Enables the creation of attributes in an existing user store.

- Redirect URL

  Sends the user to the linkaccount.jsp file when authentication fails. An authentication protects the redirect URL. The scheme requests the user to log in to create a SiteMinder session.

- Post Preservation at the Web Agent

  Must be enabled at the Service Provider Web Agent.

- Search Specification

  Indicates which attribute the NameID from the assertion replaces

Enable dynamic account linking for POST or Artifact single sign-on at the Service Provider

Follow these steps:

1. For the linkaccount.jsp file, do the following:

   - (Optional) Customize the linkaccount.jsp file to provide a custom user experience when the user is redirected after a failed authentication attempt. This file must POST the **accountlinking** and **samlresponse** parameters back to the Assertion Consumer Service URL.

     Note: The accountlinking must be set to yes (accountlinking=yes).

     The default location for this file is http://*sp_home*/affwebservices/public/.

   - Protect the linkaccount.jsp file with a SiteMinder forms authentication scheme, which supports POST-Preservation. The SAML response that contains the assertion is posted to the Assertion Consumer Service after the user has logged in locally at the Service Provider. Preserve the SAML response POST data during the entire local authentication process.

     To protect resources with an authentication scheme, refer to information about authentication schemes in the *Policy Server Configuration Guide*.

2. Enable the Allow/Create feature at the Service Provider.

3. For the Web Agent at the Service Provider, set the POST Preservation parameter to yes. This setting enables the POST data from the SAML response to be preserved.

4. Configure a redirect URL that sends the user to the linkaccount.jsp file if authentication fails. Direct the user only to this file.

   The redirect URL is part of the SAML 2.0 authentication scheme setup at the Service Provider.

Complete the following fields with the values shown:

Redirect URL for the User Not Found Status

http://*sp_home*/*protected_realm*/linkaccount.jsp

Example: http://smwidgets.com/partner_resources/linkaccount.jsp

The default location of the linkaccount.jsp file is http://*sp_home*/affwebservices/public/. Copy the file from this directory to a directory that is configured as a protected realm.

Mode

HTTP POST

5. Configure a search specification for the SAML authentication scheme. For example, if the Name ID from the assertion replaces buyerID, the search specification would be buyerID=%s.

More information:

Use Case 13: SSO with Dynamic Account Linking at the SP (see page 76)

Appendix B: Legacy Federation Process Flow

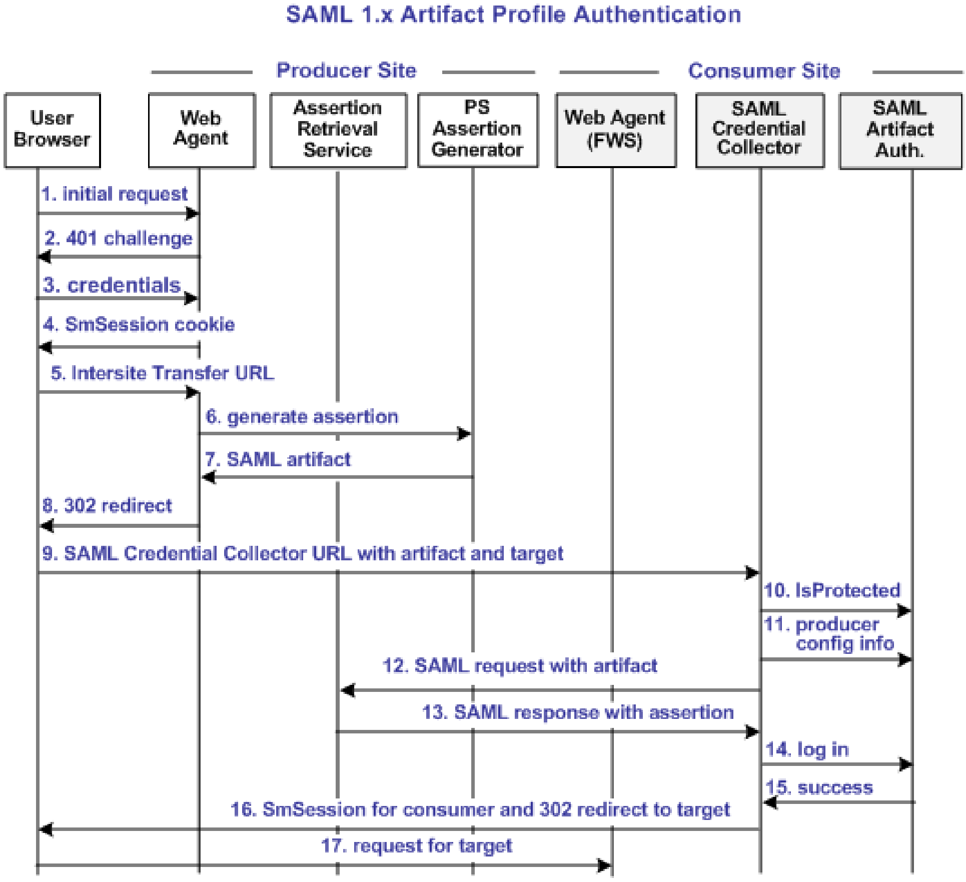
This section contains the following topics:

Flow Diagram for SSO Using SAML 1.x Artifact Authentication

The following illustration shows the flow between a user and the legacy federation components at the producer and consumer sites. This set-up enables single sign-on between the sites. SAML artifact profile is the authentication method and the flow diagram assumes successful authentication and authorization at the producer and consumer sites.

Note: This flow applies to examples that do not use the SAML Affiliate Agent.

The process flow diagram for SAML 1.x Artifact Authentication follows.



Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of actions is as follows:

1. The user makes an initial request to a protected page at the producer site.
2. The Web Agent at the producer site responds with a 401 challenge to the user.
3. The user submits credentials, such as the user name and password to the Web Agent.
4. The Web Agent issues a SiteMinder SMSESSION cookie to the browser of the user for the producer site domain.

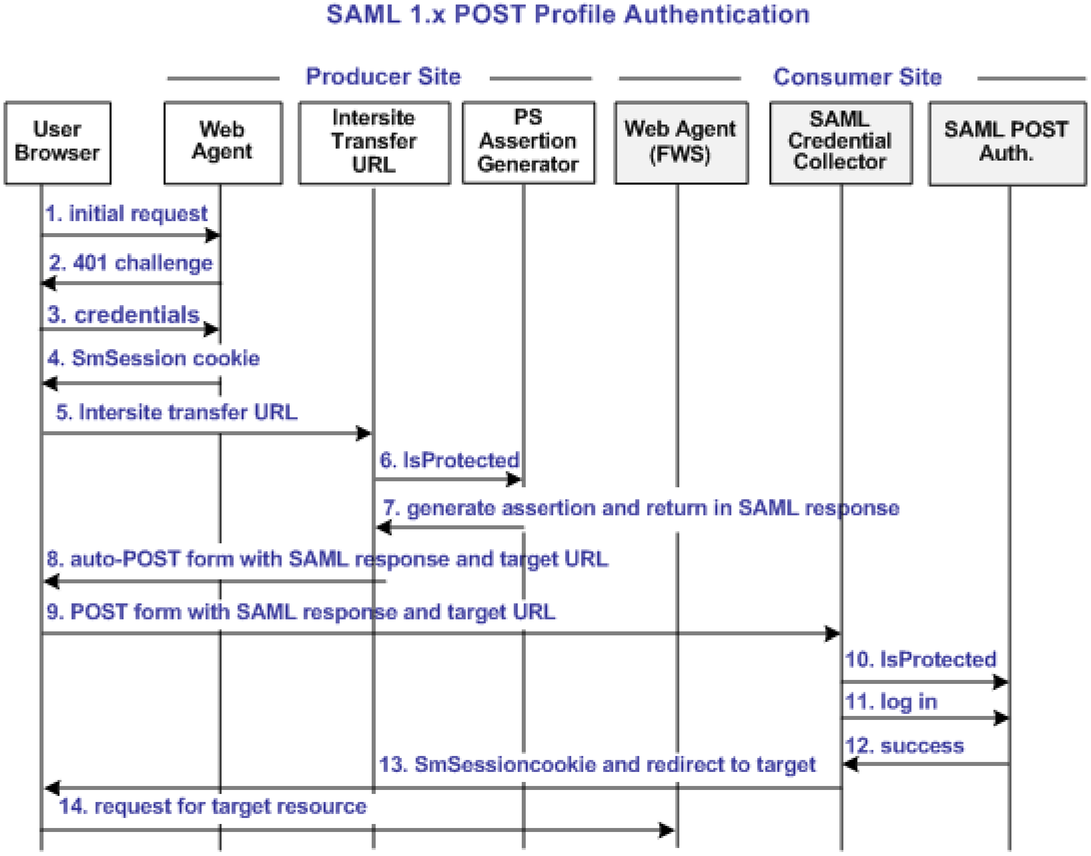
5. The user clicks a link to visit the consumer site. This link is referred to as the intersite transfer URL because it results in transferring the user to another site. The intersite transfer URL makes a request to the Web Agent at the producer site first. This URL contains the location of the SAML credential collector and the target URL to access at the consumer site.
6. The Web Agent at the producer site handles the intersite transfer URL request by calling the assertion generator.
7. The assertion generator generates a SAML assertion, places it in the session store and returns the SAML artifact for the assertion.
8. The Web Agent responds with a 302 redirect to the SAML credential collector at the consumer. The redirect contains the SAML artifact and the target URL as query parameters.
9. The browser makes a request to the URL for the SAML credential collector at the consumer site.
10. The SAML credential collector handles the URL request by making an isProtected call to the SAML artifact authentication scheme.
11. The SAML artifact authentication scheme returns the producer configuration information.
12. The SAML credential collector uses the producer configuration information to make a SAML request to the assertion retrieval service at the producer. In this step, the SAML credential collector is acting as an HTTP client.
13. The assertion retrieval service at the producer retrieves the SAML assertion from the session store. The service responds with a SAML response that contains the SAML assertion.
14. The SAML credential collector makes a login call to the SAML artifact authentication scheme, passing the SAML assertion as credentials.
15. The SAML artifact authentication scheme validates the SAML assertion. The authentication scheme looks up the user record. The lookup is based on the user mapping that is configured for the scheme. The scheme returns a success reply. If the SAML assertion is not valid or a user record cannot be located, the scheme returns a failure reply.
16. If the scheme returns a success reply, the SAML credential collector issues a SiteMinder SMSESSION cookie for the consumer domain to the browser. The SAML credential collector also issues a 302 redirect to the target URL. If the scheme returns a failure reply, the SAML credential collector issue a 302 redirect to a no access URL.
17. The browser makes a request to the target URL at the consumer, which the Web Agent protects.

Flow Diagram for SSO Using SAML 1.x POST Profile Authentication

The following illustration shows the detailed flow between a user and the components at producer and consumer sites. This set-up enables single sign-on between the sites. SAML POST profile is the authentication method and the illustration assumes successful authentication and authorization at the producer and consumer sites.

Note: This flow applies to examples that do not use the SAML Affiliate Agent.

The process flow diagram for SAML 1.x POST Profile follows.


SAML 1.x POST Profile Authentication
Producer Site
Consumer Site
User Browser
Web Agent
Intersite Transfer URL
PS Assertion Generator
Web Agent (FWS)
SAML Credential Collector
SAML POST Auth.
1. initial request
2. 401 challenge
3. credentials
4. SmSession cookie
5. Intersite transfer URL
6. IsProtected
7. generate assertion and return in SAML response
8. auto-POST form with SAML response and target URL
9. POST form with SAML response and target URL
10. IsProtected
11. log in
12. success
13. SmSessioncookie and redirect to target
14. request for target resource

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. User requests a local page at the producer, which the Web Agent protects.
2. The Web Agent at the producer asks for user credentials.

   This flow diagram assumes that the resource is protected with basic authentication and that user name and password are the required credentials.
3. The user submits credentials.
4. The Agent at the producer issues an SMSESSION cookie for the producer site domain and allows access to the local page.
5. The user selects a link at the local page of the producer to visit the consumer. The link looks like it goes to the consumer site, but it goes to the intersite transfer URL. The URL contains the affiliate name, the assertion consumer URL, and the target resource as query parameters.
6. The Intersite Transfer Service makes an IsProtected call to the Policy Server for the resource. The URL contains the name query parameter that uniquely identifies the consumer.
7. The Policy Server recognizes the request as a request for a SAML assertion. The Policy Server generates the assertion and returns it in a digitally signed SAML response message. The Policy Server then returns the response to the intersite transfer URL.
8. The intersite transfer URL service generates an auto-POST form containing the encoded SAML response and the target URL as form variables. The service sends the form to the browser.
9. The browser of the user automatically posts the HTML form to the SAML Credential Collector at the consumer site. This URL is read from the SAML response that the intersite transfer URL service sends.
10. The Credential Collector makes an isProtected call to the SAML POST profile authentication scheme. The authentication scheme informs the assertion consumer what type of credentials are required.
11. The Credential collector makes a login call for the requested target resource to the SAML POST profile authentication scheme, passing the assertion as credentials.
12. If the login succeeds, the SAML Credential Collector generates an SMSESSION cookie for the consumer site domain.
13. The SMSESSION cookie is placed in the browser and redirects the user to the target resource.
14. The browser requests the target resource, which the consumer-side Web Agent protects. The browser has an SMSESSION cookie for the consumer domain so the Web Agent does not challenge the user.

Flow Diagram for SSO Using SAML 2.0 Authentication with Artifact Binding

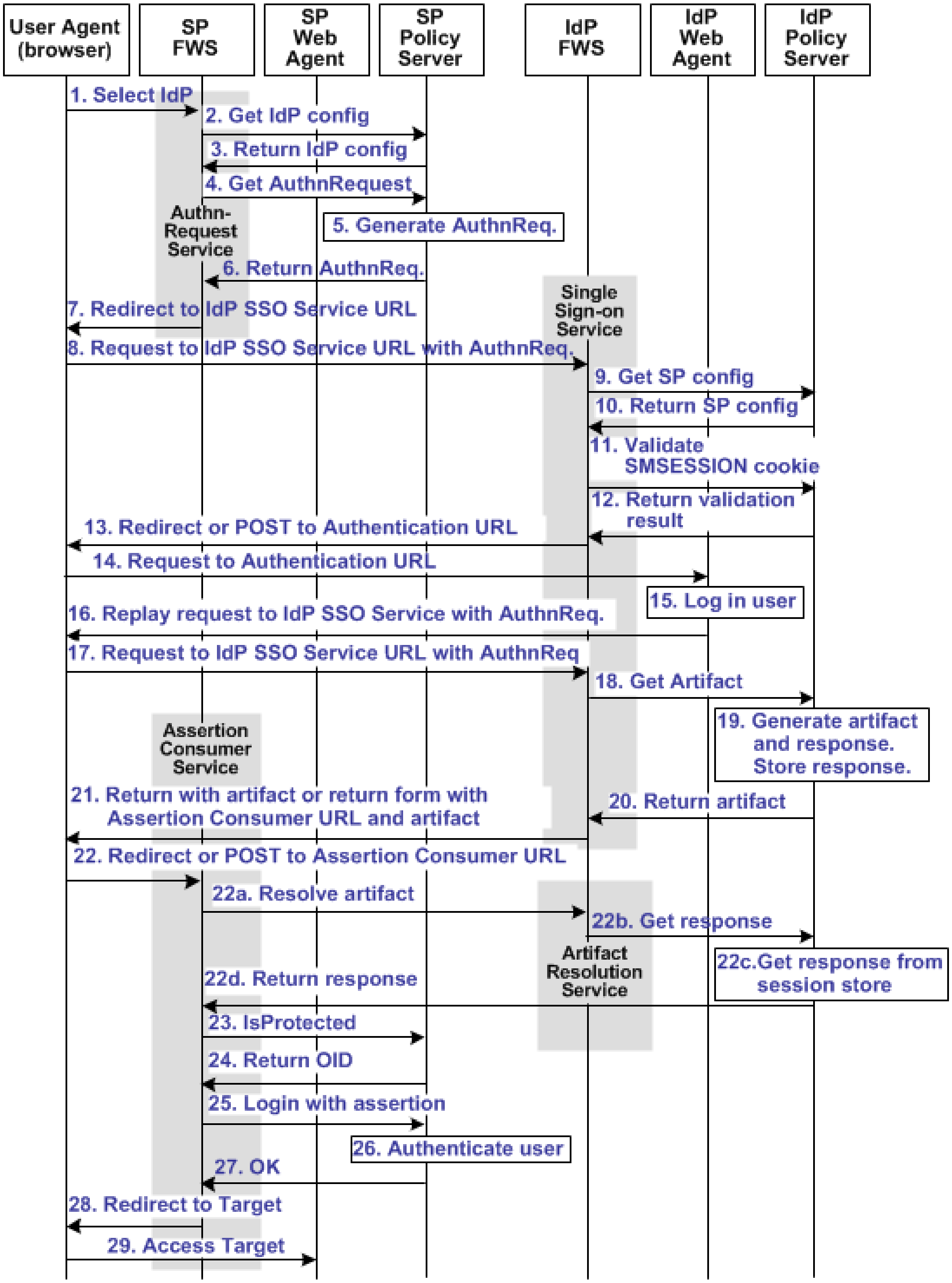
The following illustration shows the detailed flow between a user and the components at the Identity Provider and Service Provider. This set-up enables single sign-on between the sites and uses the SAML 2.0 authentication scheme with the artifact binding as the authentication method.

The flow diagram assumes the following information:

- The SP initiates the request for a resource.
- Successful authentication and authorization at the IdP and SP sites.

Note: This flow applies to examples that do not use the SAML Affiliate Agent.

The flow diagram for SAML 2.0 Authentication-Artifact Binding follows.


User Agent (browser)
SP FWS
SP Web Agent
SP Policy Server
IdP FWS
IdP Web Agent
IdP Policy Server
1. Select IdP
2. Get IdP config
3. Return IdP config
4. Get AuthnRequest
Authn-Request Service
5. Generate AuthnReq.
6. Return AuthnReq.
7. Redirect to IdP SSO Service URL
Single Sign-on Service
8. Request to IdP SSO Service URL with AuthnReq.
9. Get SP config
10. Return SP config
11. Validate SMSESSION cookie
12. Return validation result
13. Redirect or POST to Authentication URL
14. Request to Authentication URL
15. Log in user
16. Replay request to IdP SSO Service with AuthnReq.
17. Request to IdP SSO Service URL with AuthnReq
18. Get Artifact
19. Generate artifact and response. Store response.
Assertion Consumer Service
20. Return artifact
21. Return with artifact or return form with Assertion Consumer URL and artifact
22. Redirect or POST to Assertion Consumer URL
22a. Resolve artifact
22b. Get response
Artifact Resolution Service
22c.Get response from session store
22d. Return response
23. IsProtected
24. Return OID
25. Login with assertion
26. Authenticate user
27. OK
28. Redirect to Target
29. Access Target

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. The user chooses a link at the SP to authenticate at a specific IdP. This link must include a Provider ID representing the chosen IdP.
2. SP FWS requests the IdP configuration information from the local Policy Server.
3. The local Policy Server returns the IdP configuration information to SP FWS. FWS can cache the configuration information.
4. SP FWS requests an AuthnRequest message from the local Policy Server through a tunnel call, passing the Provider ID. This call must contain the artifact profile in the ProtocolBinding element value.
5. The local Policy Server generates the AuthnRequest message in an HTTP redirect binding.
6. The local Policy Server returns the AuthnRequest message to the SP FWS in an HTTP redirect binding.
7. SP FWS redirects the user to the IdP SSO Service URL. FWS obtains the configuration information and includes it in the AuthnRequest message in an HTTP redirect binding.
8. The browser requests the IdP single sign-on service (SSO) URL.
9. IdP FWS requests the SP configuration information from the IdP local Policy Server.
10. The local Policy Server returns the configuration information.

    Note: FWS can cache the configuration information.
11. IdP FWS gets an SMSESSION cookie for the domain of this IdP and calls the Policy Server to validate it. If there is no SMSESSION cookie, the IDP FWS skips redirects or posts to the Authentication URL.
12. The Policy Server verifies the validity of the SMSESSION cookie and returns the result.
13. If the SMSESSION cookie does not exist or is not valid, the IDP FWS redirects or posts to the Authentication URL. If the SMSESSION cookie is valid, the IDP FWS requests a SAML 2.0 artifact from the local Policy Server (see step 18).
14. The browser requests the Authentication URL, which the IdP Web Agent protects.
15. The IdP Web Agent logs the user in, setting the SMSESSION cookie, and lets the request pass to the Authentication URL.
16. The Authentication URL is the redirect.jsp file, which replays the request to the IdP single sign-on service with the AuthnRequest message.

17. The browser requests the IdP single sign-on service URL. This request is equivalent to the request from step 8, but now the user has a valid SMSESSION cookie.

18. The IdP FWS requests a SAML 2.0 artifact from the local Policy Server. FWS passes the AuthnRequest through an authorization call to the realm obtained from the configuration information.

19. The Policy Server generates the artifact and the corresponding response message. The message is formed from the configuration information from the Service Provider. The Policy Server stores the response in the session store.

    The message is stored as a session variable, and is named using the string representation of the artifact message handle.

20. The Policy Server returns the artifact to IdP FWS.

21. Based on the SP configuration information, the IdP FWS takes one of the following actions:

    ■ Redirects the browser to the Assertion Consumer URL at the SP. The URL-encoded artifact is a URL parameter. The Assertion Consumer URL is obtained from the configuration information.

    ■ Returns a form containing the artifact form-encoded in two hidden form controls.

      The form is wrapped into a JavaScript to auto-POST the data when the browser reads it.

    Note: The assertion generator can indicate that the authentication level for the current session is too low. If the level is too low, the IdP FWS redirects to the authentication URL to facilitate step-up authentication.

22. If the artifact was sent as part of a URL, the browser redirects the user to the Assertion Consumer URL with the artifact. If the artifact was returned in a form, then the browser POSTs the artifact to the Assertion Consumer URL.

    The following steps reflect the back-channel call that the SP FWS Assertion Consumer service makes to the IdP FWS Artifact Resolution Service to resolve the artifact into a response message.

    a. The SP FWS obtains the artifact from the GET or POST data, depending on how the IdP FWS is configured to redirect the browser. FWS then obtains the SOAP endpoint of the Artifact Resolution Service from the IdP configuration information. The source ID is part of the artifact. After the SOAP endpoint is obtained, the SP FWS makes a back-channel call to the IdP FWS Artifact Resolution service to resolve the artifact.

    b. The IdP FWS requests the response message from the local Policy Server. The message that is stored as a session variable is requested using the Java Agent API. The session ID is extracted from the artifact. The session variable name is the string representation of the artifact message handle.

c. The local Policy Server retrieves the response message from the session store and deletes it after the artifact retrieval.

d. The local Policy Server returns the response message to the IdP FWS. The IdP FWS returns the response message to the SP FWS Assertion Consumer Service.

The back-channel call is now complete.

23. The SP FWS obtains the response message from the post data. The service then determines the target resource from the configuration and makes an isProtected call to the Policy Server for the target resource.

    If the assertion is encrypted, the FWS makes a tunnel call. This call takes the encrypted assertion and returns the assertion in the clear.

24. The Policy Server returns the realm OID for the target resource.

25. The SP FWS passes the response message to the local Policy Server through a login call. The response message acts as the credentials and the realm OID is obtained from the isProtected call.

26. The SAML 2.0 authentication scheme logs the user in using the response message as credentials.

27. The local Policy Server returns OK to the SP FWS.

28. If a success reply is returned, SP FWS creates an SMSESSION cookie for the SP domain. The service places the cookie in the browser and redirects the user to the target URL, which is obtained from the configuration information.

    If the login fails, the SP FWS redirects the user to a No Access URL.

29. The browser of the user requests the target URL, which the SP-side Web Agent protects. Because the browser has an SMSESSION cookie, the Web Agent does not challenge the user.

Flow Diagram for SSO Using SAML 2.0 Authentication with POST Binding

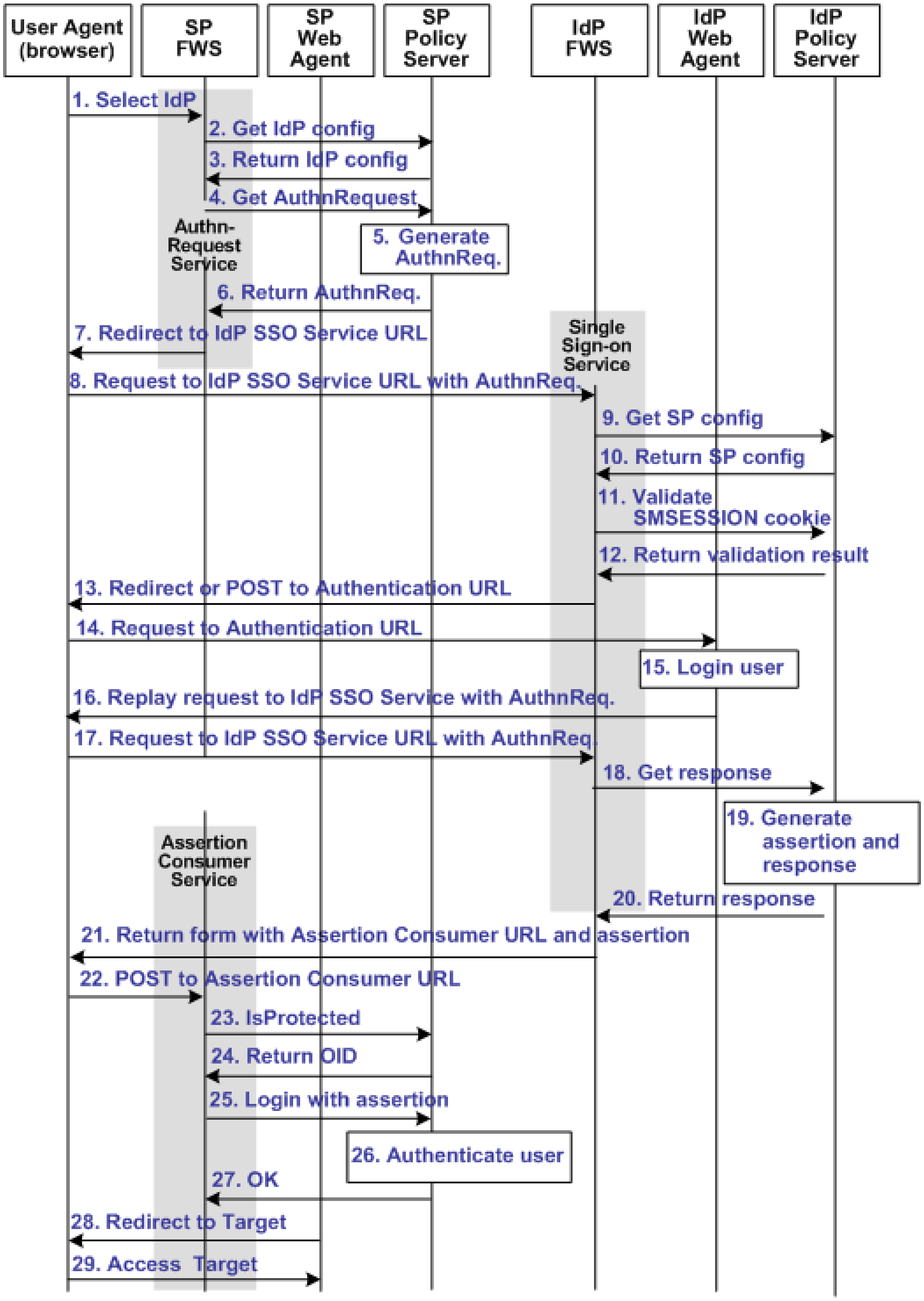
The following illustration shows the detailed flow between a user and the components deployed at an Identity Provider (IdP) and Service Provider (SP) sites. This set-up enables single sign-on between the sites, using SAML 2.0 POST binding as the method of obtaining the SAML assertion for authentication.

The flow diagram assumes the following:

- The SP initiates the request for a resource.
- Successful authentication and authorization at the IdP and SP sites.

Note: This flow applies to examples that do not use the SAML Affiliate Agent.

The flow diagram for SAML 2.0 authentication-POST binding follows.


User Agent (browser)
SP FWS
SP Web Agent
SP Policy Server
IdP FWS
IdP Web Agent
IdP Policy Server
1. Select IdP
2. Get IdP config
3. Return IdP config
4. Get AuthnRequest
Authn-Request Service
5. Generate AuthnReq.
6. Return AuthnReq.
7. Redirect to IdP SSO Service URL
Single Sign-on Service
8. Request to IdP SSO Service URL with AuthnReq.
9. Get SP config
10. Return SP config
11. Validate SMSESSION cookie
12. Return validation result
13. Redirect or POST to Authentication URL
14. Request to Authentication URL
15. Login user
16. Replay request to IdP SSO Service with AuthnReq.
17. Request to IdP SSO Service URL with AuthnReq.
18. Get response
19. Generate assertion and response
Assertion Consumer Service
20. Return response
21. Return form with Assertion Consumer URL and assertion
22. POST to Assertion Consumer URL
23. IsProtected
24. Return OID
25. Login with assertion
26. Authenticate user
27. OK
28. Redirect to Target
29. Access Target

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. The user chooses a link at the SP to authenticate at a specific IdP. This link must include a Provider ID representing the chosen IdP.
2. SP FWS requests the IdP configuration information from the local Policy Server.
3. The Policy Server returns the IdP configuration information to SP FWS. FWS can cache this configuration information.
4. SP FWS requests an AuthnRequest message from the local Policy Server through a tunnel call, passing the Provider ID.
5. The Policy Server generates the AuthnRequest message in an HTTP redirect binding.
6. The local Policy Server returns the AuthnRequest message to the SP FWS in an HTTP redirect binding.
7. SP FWS redirects the user to the IdP Single Sign-on Service URL, which is obtained from the configuration information with the AuthnRequest message.
8. The browser requests the IdP Single Sign-on Service URL.
9. IdP FWS requests the SP configuration information from the IdP local Policy Server.
10. The local Policy Server returns the configuration information.

    Note: FWS can cache the configuration information.
11. IdP FWS gets an SMSESSION cookie for this domain of the IdP and calls the Policy Server to validate it. If there is no SMSESSION cookie, the IDP FWS redirects or posts to the Authentication URL.
12. The Policy Server verifies the validity of the SMSESSION cookie and returns the result.
13. If the SMSESSION cookie does not exist or is not valid, the IDP FWS redirects or posts to the Authentication URL. FWS obtains this URL from the configuration information. If the SMSESSION cookie is valid, the IDP FWS skips to 18.
14. The browser requests the Authentication URL, which the IdP Web Agent protects.
15. The IdP Web Agent logs the user in, setting the SMSESSION cookie and lets the request pass to the Authentication URL.
16. The Authentication URL is the redirect.jsp file, which replays the request to the IdP single sign-on service with the AuthnRequest message.
17. The browser requests the IdP single sign-on service URL. This request is equivalent to the request from step 8, but now the user has a valid SMSESSION cookie.

18. The IdP FWS requests a SAML 2.0 assertion from the Policy Server. The AuthnRequest goes through an authorize call to the realm obtained from the configuration information.

19. The Policy Server generates an assertion that is based on the configuration information for the SP, signs it, and returns the assertion wrapped in a response message.

20. The response message is returned to IdP FWS.

21. IdP FWS returns a form to the user. The form contains the response message, the Assertion Consumer URL, obtained from the configuration information, and the JavaScript to submit the form.

    Note: If the assertion generator indicates that the current sessions authentication level too low, the IdP FWS redirects to the authentication URL as in Step 13 to facilitate step-up authentication.

22. The browser posts the response message to the Assertion Consumer URL at the SP.

23. The SP FWS obtains the response message from the POST data. FWS then determines the target resource from the configuration and makes an isProtected call to the Policy Server for the target resource.

    If the assertion is encrypted, the FWS makes a tunnel call. The call takes the encrypted assertion and returns the assertion in the clear.

24. The Policy Server returns the realm OID for the target resource.

25. The SP FWS passes the response message to the local Policy Server through a login call. FWS uses the response message as credentials and the realm OID obtained from the isProtected call.

26. The SAML 2.0 authentication scheme logs the user in using the response message as credentials.

27. The local Policy Server returns OK to the SP FWS.

28. If a success reply is returned, SP FWS creates an SMSESSION cookie for the SP domain. FWS then places the cookie in the browser and redirects the user to the target URL, which is obtained from the configuration information.

    If the login fails, the SP FWS redirects the user to a No Access URL.

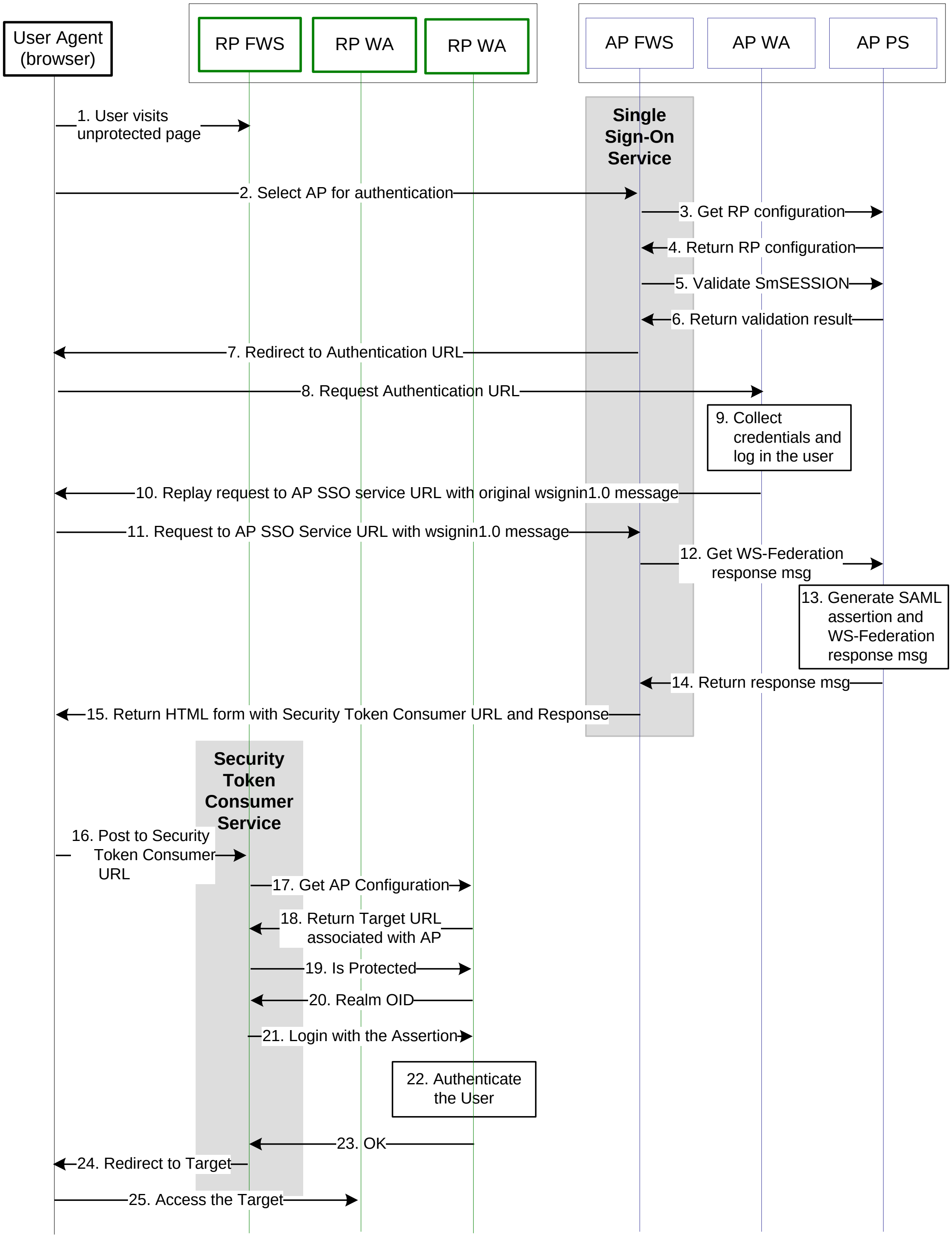
29. The browser sends a request to the target URL, which the SP-side Web Agent protects. Because the browser has an SMSESSION cookie, the Web Agent does not challenge the user.

Flow Diagram for WS-Federation SSO Initiated at the Resource Partner

The following illustration shows the detailed flow between a user and the legacy federation components at an Account Partner (AP) and Resource Partner (RP) sites. This set-up enables single sign-on between the sites, using WS-Federation as the method of obtaining the SAML assertion for authentication.

The flow diagram assumes the following information

- The Resource Partner initiates the request for a resource.
- Successful authentication and authorization at the AP and RP sites.


User Agent (browser)
RP FWS
RP WA
RP WA
AP FWS
AP WA
AP PS
Single Sign-On Service
1. User visits unprotected page
2. Select AP for authentication
3. Get RP configuration
4. Return RP configuration
5. Validate SmSESSION
6. Return validation result
7. Redirect to Authentication URL
8. Request Authentication URL
9. Collect credentials and log in the user
10. Replay request to AP SSO service URL with original wsignin1.0 message
11. Request to AP SSO Service URL with wsignin1.0 message
12. Get WS-Federation response msg
13. Generate SAML assertion and WS-Federation response msg
14. Return response msg
15. Return HTML form with Security Token Consumer URL and Response
Security Token Consumer Service
16. Post to Security Token Consumer URL
17. Get AP Configuration
18. Return Target URL associated with AP
19. Is Protected
20. Realm OID
21. Login with the Assertion
22. Authenticate the User
23. OK
24. Redirect to Target
25. Access the Target

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The WS-Federation single sign-on process is as follows:

1. The user visits an unprotected site selection page at the Resource Partner.
2. The user chooses a link to authenticate for AP that is a federated partner. This link points to the Single Sign-on Service at the AP. The link must contain the Provider ID of the RP and can include some optional parameters, such as the wctx parameter. The browser requests the AP SSO Service URL.
3. Based on RP provider ID specified as a query parameter, the AP FWS requests the RP configuration information from the local Policy Server.
4. The local Policy Server returns the configuration information.

   Note: The FWS can cache the configuration information.
5. The AP FWS gets the SMSESSION cookie for the AP domain and calls the Policy Server to validate it. If there is no SMSESSION cookie, the AP FWS skips to step 7.
6. The Policy Server verifies the validity of the SMSESSION cookie and returns the result to the FWS application.
7. If the SMSESSION cookie does not exist or is not valid, the AP FWS redirects the user to the Authentication URL obtained from the RP configuration information. If the SMSESSION cookie is valid, the AP FWS skips to step 12.
8. The browser requests the Authentication URL, which the AP Web Agent protects.
9. The AP WA authenticates the user and sets the SMSESSION cookie. The AP WA lets the request pass to the Authentication URL.
10. The Authentication URL points to the redirect.jsp, which replays the request to the AP SSO service with the original wsignin message.
11. The browser requests the AP SSO Service URL. This request is equivalent to the request from step 2, but now the user has a valid SMSESSION cookie.
12. The AP FWS requests a WS-Federation <RequestSecurityTokenResponse> from the Policy Server through an authorize call to the realm obtained from the configuration information.
13. The Policy Server generates a SAML1.1 assertion that is based on the configuration information for the RP. The Policy Server signs the assertion and returns it wrapped in an <RequestSecurityTokenResponse> message.
14. The <RequestSecurityTokenResponse> message is returned to the AP FWS.

15. The AP FWS returns a form to the user containing the following information:

    - URL encoded <RequestSecurityTokenResponse> message.
    - Security Token Consumer Service URL.
    - Optional wctx that came with the wsignin message.
    - JavaScript to auto submit the form.

    If the original wsignin request contains the wreply parameter, its value becomes the Security Token Consumer URL. The wreply value becomes the URL only if the Security Token Consumer URL setting is not in the RP configuration information. For security reasons, the Security Token Consumer URL setting in the RP configuration information takes precedence over the wreply parameter.

    Note: The assertion generator can indicate that the authentication level of the current session is too low. If the level is too low, the AP FWS redirects to the authentication URL as in step 7 to facilitate “step-up” authentication.

16. The user agent posts the <RequestSecurityTokenResponse> message and wctx to the Security Token Consumer URL at the RP.
17. The RP FWS obtains the <RequestSecurityTokenResponse> message and wctx from the POST data. RP FWS requests the AP configuration information from the local Policy Server.
18. RP FWS determines the target resource from the AP configuration information from local Policy Server. If the target resource is not part of the AP configuration, and the wctx parameter is found in the POST data, the wctx value becomes the target resource.
19. FWS makes an isProtected call to the Policy Server for the target resource.
20. The Policy Server returns the realm OID for the target resource.
21. The RP FWS passes the <RequestSecurityTokenResponse> message to the local Policy Server through a login call. The <RequestSecurityTokenResponse> message and the realm OID obtained from the isProtected call service as credentials.
22. The WS-Federation authentication scheme logs the user in using the <RequestSecurityTokenResponse> message as credentials.
23. The local Policy Server returns an OK status message to the RP FWS.
24. The RP FWS creates the SMSESSION cookie for the RP domain. FWS places the cookie in the browser and redirects the user to the Target URL or to the wctx POST data. If the login fails, the RP FWS redirects the user to a No Access URL.
25. The user agent requests the Target URL that the RP-side Web Agent protects. Because the browser has the SMSESSION cookie for the RP domain, the Web Agent does not have to challenge the user.

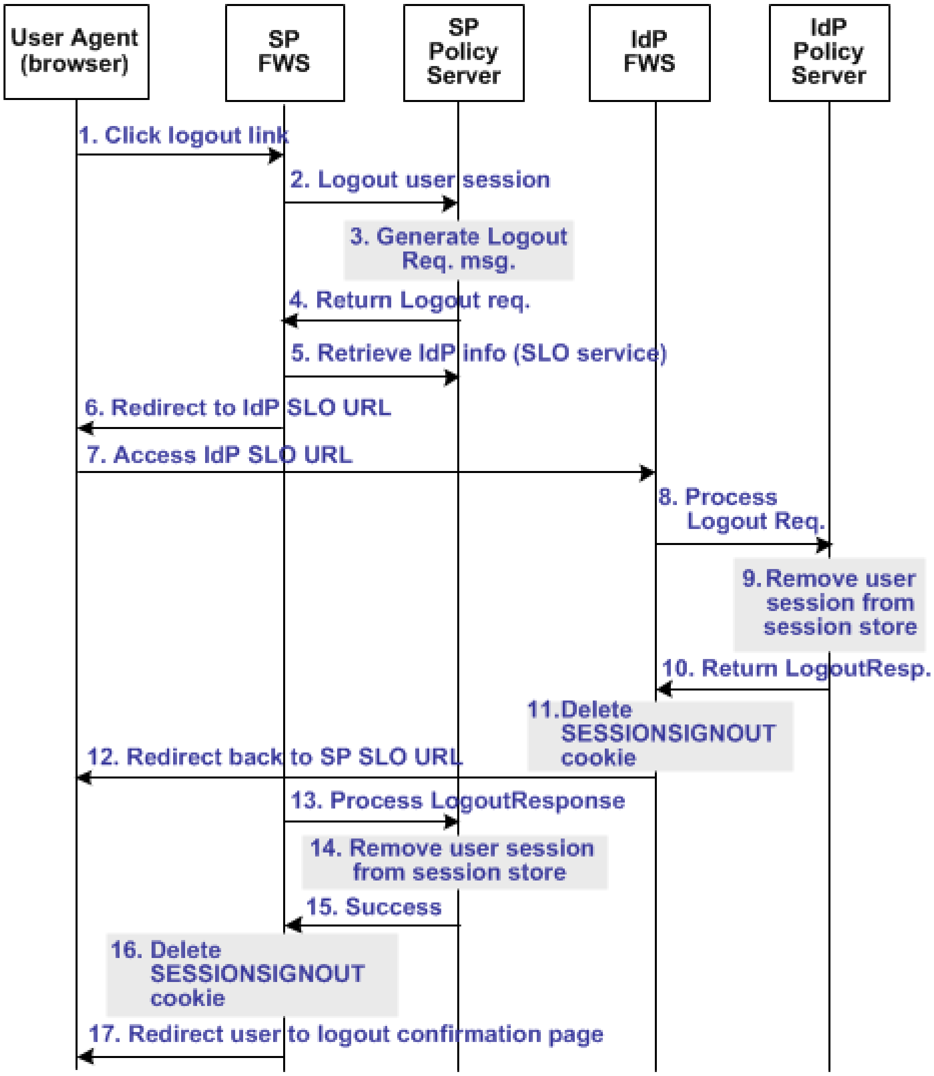
WS-Federation SSO Initiated at the Account Partner

Single sign-on that is initiated by the Account Partner is similar to the RP-initiated use case. HTML content at AP contains intersite transfer links to different RP sites. When the user clicks any link, the web browser requests the AP SSO Service URL. The rest of the processing is same as specified in the RP-initiated use case.

Flow Diagram for SAML 2.0 Single Logout

The following illustration shows the detailed flow for a single logout request between a user and the legacy federation components at an Identity Provider (IdP) and Service Provider (SP). This set-up enables single logout for all entities that have a session with a particular user.

The following illustration assumes that the SP initiates the log out request.


User Agent (browser)
SP FWS
SP Policy Server
IdP FWS
IdP Policy Server
1. Click logout link
2. Logout user session
3. Generate Logout Req. msg.
4. Return Logout req.
5. Retrieve IdP info (SLO service)
6. Redirect to IdP SLO URL
7. Access IdP SLO URL
8. Process Logout Req.
9. Remove user session from session store
10. Return LogoutResp.
11.Delete SESSIONSIGNOUT cookie
12. Redirect back to SP SLO URL
13. Process LogoutResponse
14. Remove user session from session store
15. Success
16. Delete SESSIONSIGNOUT cookie
17. Redirect user to logout confirmation page

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack, which provide the FWS application functions. For more information about installing and configuring the SPS federation gateway, see the *CA SiteMinder Secure Proxy Server Administration Guide*.

The sequence of events is as follows:

1. The user clicks a link at SP to end the global session. The browser of the user accesses the Single Logout servlet at the SP.

   SP FWS renames the SMSESSION cookie to SESSIONSIGNOUT to invalidate the current session of the user.

2. FWS reads the SessionId value from the SESSIONSIGNOUT cookie and asks the Policy Server to terminate the user session.

3. Based on the session store information, the user session status is changed to a LogoutInProgress state in the session store. The Policy Server determines that the user session is created based on the SAML assertion received from an IdP. The Policy Server generates a LogoutRequest request to invalidate the user session at the IdP.

4. The Policy Server returns a LogoutRequest request to SP FWS. The Policy Server also returns the Provider ID of the IdP and provider type.

5. SP FWS retrieves the provider configuration data of the IdP, which includes the SLO service URL, from the Policy Server.

6. SP FWS redirects the user to the SLO service at the IdP with the SAML LogoutRequest message added as a query parameter.

7. The browser of the user accesses SLO service at the IdP.

   When the IdP FWS receives a LogoutRequest message, it renames the SMSESSION cookie to SESSIONSIGNOUT.

8. The IdP processes the signed LogoutRequest message. The IdP then tries to invalidate the user session at all SPs specified in the session store for that session. The only SP that is not invalidated is the SP that sent the original LogoutRequest.

   Note: The process for logging the user out at each SP is similar to Step 2 through Step 7.

9. After terminating the user session from all relevant SPs, the IdP removes the user session from the session store.

10. The IdP Policy Server returns a signed LogoutResponse message to the IdP FWS, containing the provider ID of the SP and provider type. The IdP Policy Server also informs FWS that the user session is removed from the session store.

11. After learning that the user session is removed from the session store, IdP FWS deletes the SESSIONSIGNOUT cookie.

12. The IdP FWS redirects the user to the single logout service at the SP with the SAML LogoutResponse message added as a query parameter. The single logout service is part of the SP FWS application.

    The browser of the user accesses SLO service of the SP, which processes the signed LogoutResponse message.

    If the LogoutResponse message contains non-SUCCESS return code, FWS issues a SIGNOUTFAILURE cookie, and a base 64-encoded Partner ID is appended to the cookie value. If there are multiple IDs in the cookie, a space character separates them.

13. The SP Policy Server receives the LogoutResponse message from FWS and processes it.

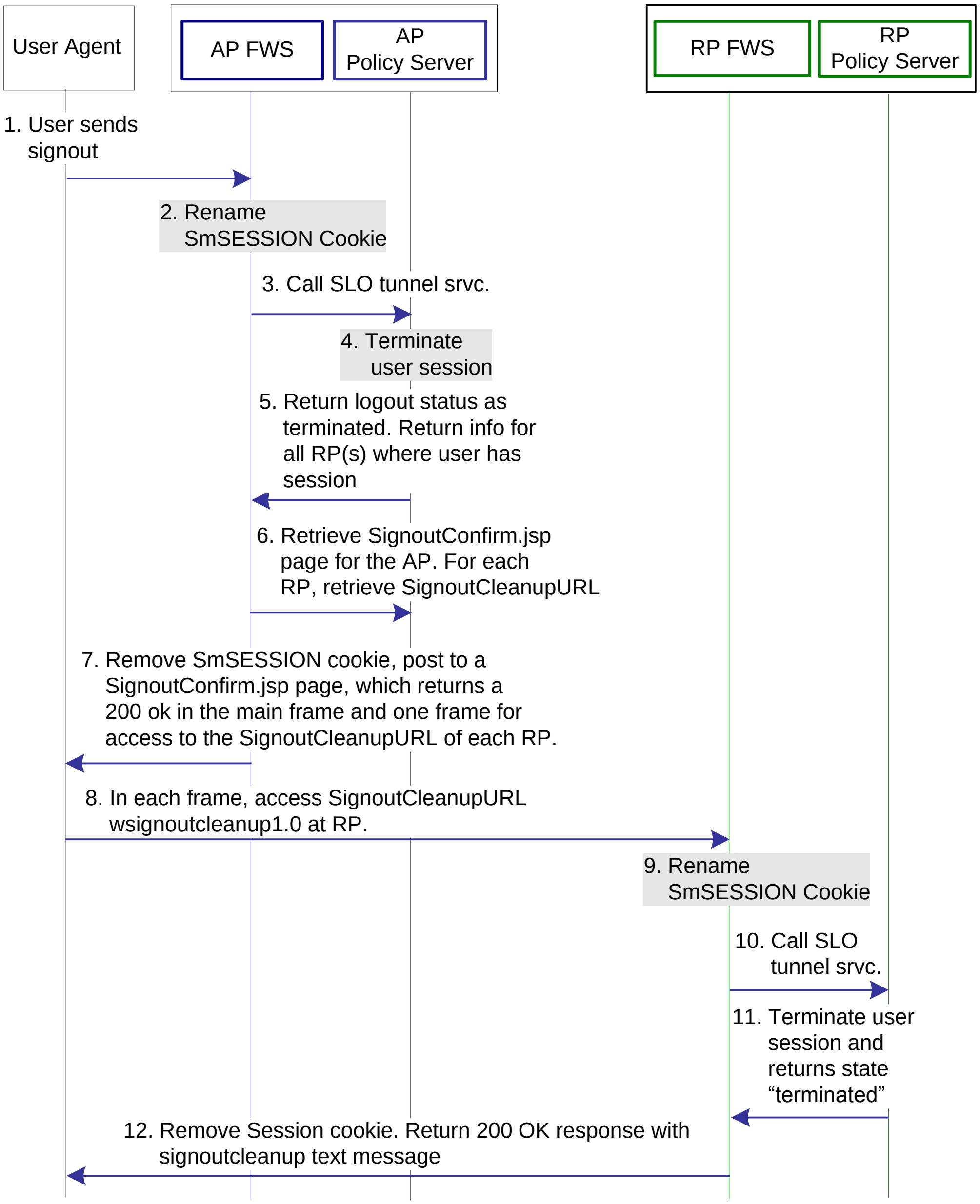
14. The SP Policy Server removes the user session from the session store.

15. After the session is removed from the session store, the Policy Server sends a SUCCESS return code to FWS. The Policy Server includes the SP ID in the final LogoutResponse message.
16. If there are no more LogoutRequest or LogoutResponse messages to process, SP FWS deletes the SESSIONSIGNOUT cookie.
17. FWS redirects the user to the Logout Confirmation page at the SP.

Flow Diagram for WS-Federation Signout (AP-initiated)

The following illustration shows the flow for a signout request between a user and the legacy federation components deployed at an Account Partner (AP) and Resource Partner. This set-up enables signout for all entities that have a session with a particular user.

The following illustration assumes that the AP initiates the signout request.


User Agent
AP FWS
AP Policy Server
RP FWS
RP Policy Server
1. User sends signout
2. Rename SmSESSION Cookie
3. Call SLO tunnel srvc.
4. Terminate user session
5. Return logout status as terminated. Return info for all RP(s) where user has session
6. Retrieve SignoutConfirm.jsp page for the AP. For each RP, retrieve SignoutCleanupURL
7. Remove SmSESSION cookie, post to a SignoutConfirm.jsp page, which returns a 200 ok in the main frame and one frame for access to the SignoutCleanupURL of each RP.
8. In each frame, access SignoutCleanupURL wsignoutcleanup1.0 at RP.
9. Rename SmSESSION Cookie
10. Call SLO tunnel srvc.
11. Terminate user session and returns state “terminated”
12. Remove Session cookie. Return 200 OK response with signoutcleanup text message

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack, which provide the FWS application functions. For more information about installing and configuring the SPS federation gateway, see the *CA SiteMinder Secure Proxy Server Administration Guide*.

When signout is initiated at the Account Partner, the process flow is as follows:

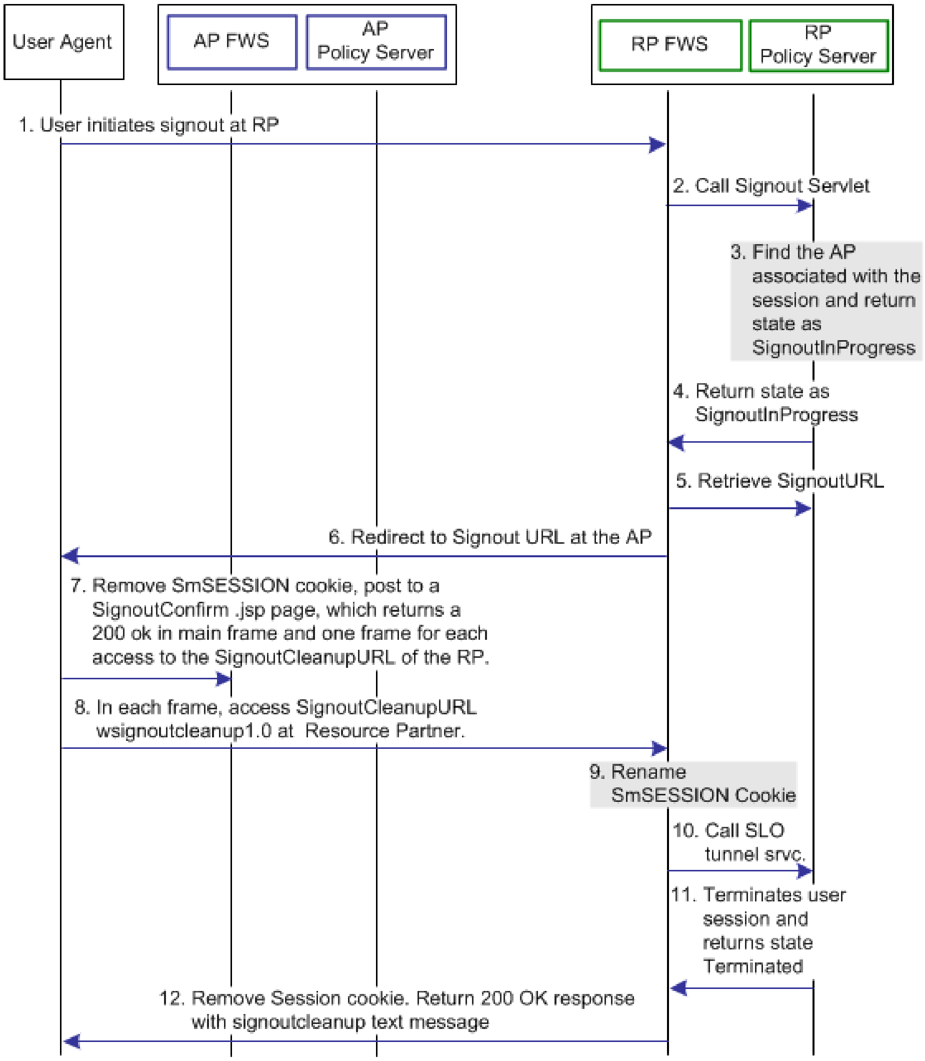
1. The user clicks on a link at the Account Partner to end the global session. The browser of the user sends a HTTP-based wsignout request to the signout servlet at the Account Partner.
2. FWS renames the SMSESSION cookie to SESSIONSIGNOUT to invalidate the current session of the user.
3. FWS reads the SessionId value from the SESSIONSIGNOUT cookie and calls the SLO Tunnel Service API to terminate the user session from the session store.
4. The SLO Tunnel Service API sets the user session status to "Terminated" in the session store. The service also removes all the RP references from the session store that are associated with that user session.
5. The SLO Tunnel Service API returns the logout status "Terminated" to the FWS Signout Servlet. The Tunnel library also returns the RP providerID and providerType for all the RPs associated with the user session.
6. FWS retrieves the provider configuration data of the RP, which includes the signout cleanup URL, from the cache of the provider maintained in FWS.
7. FWS removes the SESSIONSIGNOUT cookie then posts an AP Signout message and multiple RP-SignoutCleanup locations as post data to the SignoutConfirmURL JSP. The SignoutConfirmURL JSP is responsible for parsing various post variables and creating a frame-based HTML page. The mainframe in this HTML page displays the AP-SignOut message. Each of the remaining frames accesses the SignoutCleanupURL of individual RPs associated with the user session.
8. The browser of the user accesses SignoutCleanup service at the Resource Partner site in an individual frame.
9. When the RP FWS (Signout Servlet) receives a wsignoutcleanup request, it renames the SMSESSION cookie to SESSIONSIGNOUT. FWS then calls the SLO Tunnel Service API to process the wsignoutcleanup request.
10. The SLO tunnel library processes the wsignoutcleanup request and terminates the user session from the session store.
11. Then SLO tunnel library returns FWS with a "Terminated" status message indicating that the user session no longer exists in the session store.
12. The FWS Signout Servlet removes the SESSIONSIGNOUT cookie and returns a 200 OK response in the frame.

Note: Steps 8-12 are repeated for individual RPs simultaneously in different frames of the same HTML page.

Flow Diagram for WS-Federation Signout (RP-initiated)

The following illustration shows the flow for a signout request between a user and the legacy federation components at an Account Partner (AP) and Resource Partner. This set-up enables signout for all entities that have a session with a particular user.

The following illustration assumes that the RP initiates the sign out request.


User Agent
AP FWS
AP Policy Server
RP FWS
RP Policy Server
1. User initiates signout at RP
2. Call Signout Servlet
3. Find the AP associated with the session and return state as SignoutInProgress
4. Return state as SignoutInProgress
5. Retrieve SignoutURL
6. Redirect to Signout URL at the AP
7. Remove SmSESSION cookie, post to a SignoutConfirm .jsp page, which returns a 200 ok in main frame and one frame for each access to the SignoutCleanupURL of the RP.
8. In each frame, access SignoutCleanupURL wsignoutcleanup1.0 at Resource Partner.
9. Rename SmSESSION Cookie
10. Call SLO tunnel srvc.
11. Terminates user session and returns state Terminated
12. Remove Session cookie. Return 200 OK response with signoutcleanup text message

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack, which provide the FWS application functions. For more information about installing and configuring the SPS federation gateway, see the *CA SiteMinder Secure Proxy Server Administration Guide*.

When signout is initiated at the Resource Partner, the process flow is as follows:

1. The user clicks a link at the Resource Partner to end the global session. The browser sends a HTTP-based wsignout request to the Signout servlet at the Resource Partner.

   Note: The RP site is receiving a wsignout message and not a wsignoutcleanup message.

2. FWS reads the SessionId value from the SMSESSION cookie, renames the SMSESSION cookie to SESSIONSIGNOUT, and calls the SLO tunnel library with the wsignout request.

3. Based on information in the session store, the tunnel library determines determines the AP associated with the user session. The SLO tunnel library sets the user session state to SignoutInProgress, but does not terminate it.

4. The tunnel library returns the SignoutInProgress state message and the Account Partner providerID and providerType.

5. FWS retrieves Account Partner configuration data, which includes the Signout URL, from the FWS cache or Policy Server.

6. FWS redirects the browser of the user to the Signout URL.

7. FWS removes the SESSIONSIGNOUT cookie then posts an AP signout message and multiple RP-SignoutCleanup locations as post data to the SignoutConfirmURL JSP. The SignoutConfirmURL JSP is responsible for parsing various post variables and creating a frame-based HTML page. The primary frame in this HTML page displays the AP-SignOut message. Each of the remaining frames accesses the SignoutCleanupURL of individual RPs associated with the user session.

8. The browser accesses SignoutCleanup service at the Resource Partner site in an individual frame.

9. When RP FWS (Signout Servlet) receives a wsignoutcleanup request, it renames the SMSESSION cookie to SESSIONSIGNOUT. The service then calls the SLO Tunnel Service API to process the wsignoutcleanup request.

10. The SLO tunnel library processes the wsignoutcleanup request and terminates the user session from the session store.

11. Then SLO tunnel library returns FWS with a Terminated status message indicating that the user session no longer exists in the session store.

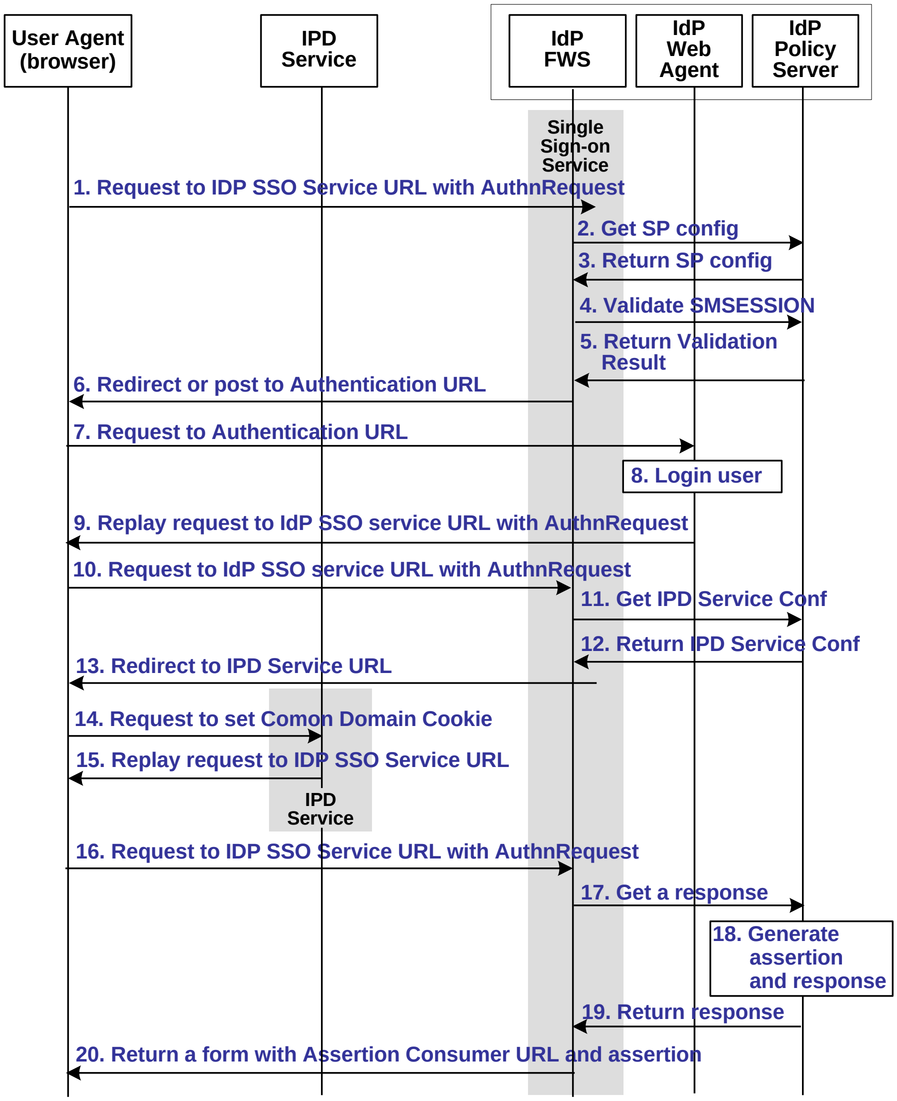
12. The FWS Signout Servlet removes the SESSIONSIGNOUT cookie and returns a 200 OK response in the frame.

Note: Steps 8-12 are repeated for individual RPs simultaneously in different frames of the same HTML page.

Flow Diagram for Identity Provider Discovery Profile

The following illustration shows the flow for an Identity Provider Discovery service between the user and the legacy federation components at an Identity Provider. This set-up involves redirecting from an Identity Provider to the Identity Provider Discovery Profile service to set the common domain cookie.

The following illustration assumes that the SP FWS redirects the user to the IdP SSO Service URL.


User Agent (browser)
IPD Service
IdP FWS
IdP Web Agent
IdP Policy Server
Single Sign-on Service
1. Request to IDP SSO Service URL with AuthnRequest
2. Get SP config
3. Return SP config
4. Validate SMSESSION
5. Return Validation Result
6. Redirect or post to Authentication URL
7. Request to Authentication URL
8. Login user
9. Replay request to IdP SSO service URL with AuthnRequest
10. Request to IdP SSO service URL with AuthnRequest
11. Get IPD Service Conf
12. Return IPD Service Conf
13. Redirect to IPD Service URL
14. Request to set Comon Domain Cookie
15. Replay request to IDP SSO Service URL
IPD Service
16. Request to IDP SSO Service URL with AuthnRequest
17. Get a response
18. Generate assertion and response
19. Return response
20. Return a form with Assertion Consumer URL and assertion

Note: The SPS federation gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions. In the flow diagram, the Web Agent block would be the embedded Web Agent in the SPS federation gateway. For information about installing and configuring the SPS federation gateway, see the *Secure Proxy Server Administration Guide*.

The Identity Provider Discovery process is as follows:

1. The user agent (browser) requests the IdP SSO Service URL.
2. The IdP FWS requests the SP configuration information from the local Policy Server.
3. The local Policy Server returns the configuration information.

   Note: The FWS can cache the configuration information.
4. The IdP FWS gets the SMSESSION cookie for the IdP domain and calls to the Policy Server to validate it. If there is no SMSESSION cookie, the IdP FWS skips to Step 6.
5. The Policy Server verifies the validity of the SMSESSION cookie and returns the result.
6. If the SMSESSION cookie does not exist or is not valid, the IdP FWS redirects or posts to the Authentication URL obtained from the configuration. If the SMSESSION cookie is valid, the IdP FWS skips to Step 18.
7. The user agent requests the Authentication URL. The IdP Web Agent protects the Authentication URL.
8. The IdP Web Agent logs the user in, setting the SMSESSION cookie and lets the request pass to the Authentication URL.
9. The Authentication URL is the redirect.jsp file, which replays the request to the IdP SSO Service with the AuthnRequest message.
10. The user agent requests the IdP SSO Service URL. This request is equivalent to the request from step 8, but now the user has a valid SMSESSION cookie.
11. The IdP FWS requests the Identity Provider Discovery Profile (IPD) configuration from the Policy Server, passing the Identity Provider ID.
12. The Policy Server returns with the IPD configuration, such as IPD Service URL, common domain cookie, and persistence information of the common domain cookie.
13. The IdP FWS redirects the user to the IPD Service URL to set the common domain cookie.
14. The IdP FWS redirects the user to the IPD Service URL.
15. The IPD Service sets or updates the common domain cookie with the Identity Provider ID. The IPD Service redirects the user agent back to the IdP FWS from which it received the Set Request.
16. The user agent requests the IdP SSO Service URL.

17. The IdP FWS requests a SAML 2.0 assertion from the Policy Server, passing the AuthnRequest through an authorize call to the realm obtained from the configuration.
18. The Policy Server generates an assertion that is based on the configuration information for the Service Provider. The Policy Server signs the assertion and returns the assertion wrapped in a response message.
19. The response message is returned to the IdP FWS.
20. The IdP FWS returns a form to the user containing the response message, the Assertion Consumer URL obtained from the configuration and Javascript to submit the form.

    Note: The assertion generator can indicate that the authentication level of the current session is too low. If the level is too low, the IdP FWS redirects to the authentication URL as in Step 13 to facilitate step-up authentication.

After the final step in the diagram, the user agent posts the response message to the Assertion Consumer URL at the Service Provider.

Index