CA Single Sign-On - 12.52 SP1

Implementing

Date: 13-Jul-2017

Table of Contents

Implementing

The content in this section provides architectural and configuration information to consider when deploying CA Single Sign-On, CA Single Sign-On Federation, and CA Access Gateway. Use the Table of Contents to access the content.

Implementing CA Single Sign-On

The content in this section provides architectural and configuration information to use when planning a new CA Single Sign-On implementation or modifying an existing implementation.

Use the Table of Contents to access the content.

Architectural Use Cases

Contents

The purpose of the following use cases is to get you thinking about your CA Single Sign-On architecture in terms of high availability and performance. The use cases begin with a simple deployment and progress into more complex scenarios. Each case is based on the idea of a logical "block" of CA Single Sign-On components and illustrates how an environment can contain multiple blocks to address the following architectural considerations:

- Redundancy
- Failover
- Capacity and scale
- Multiple cookie domains

Extrapolate the necessary infrastructure from these cases to:

- Determine how to implement redundancy and high availability between CA Single Sign-On components
- Determine how to implement multiple data centers
- Support the usage metrics you gather from capacity planning
- Support your implementation considerations
- Begin the iterative process of performance tuning the environment

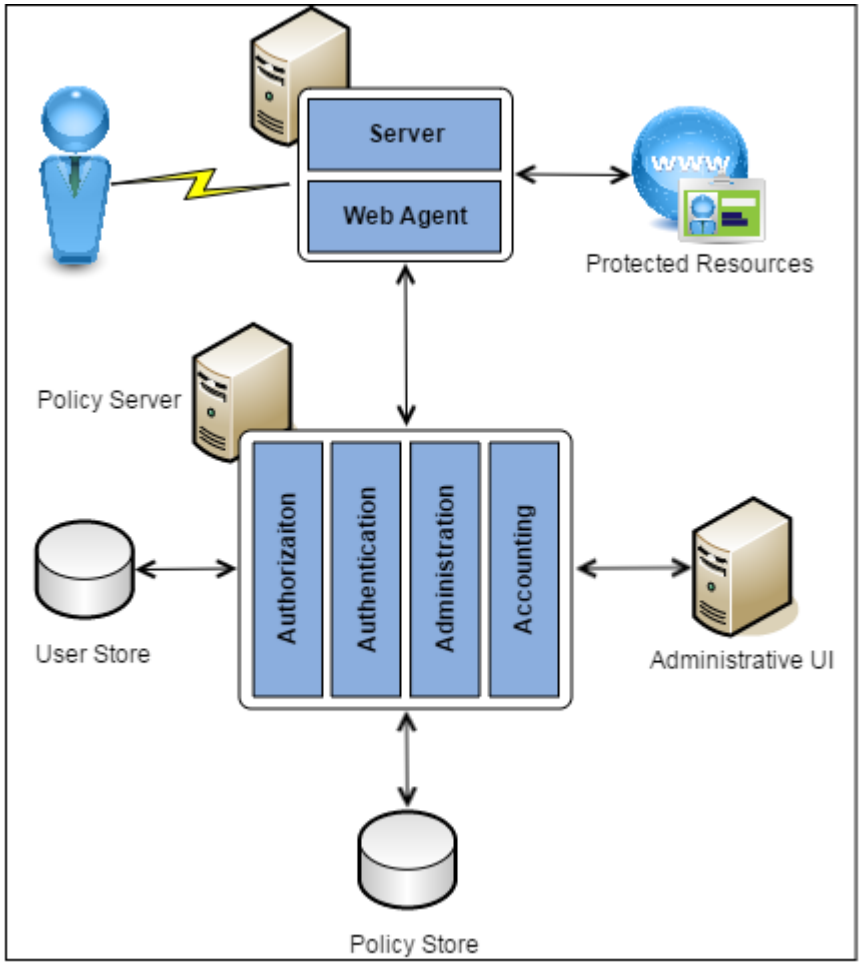
Simple Deployment

The simplest deployment requires one "block" of components. A block of components is a logical combination of dependent components that include:

- A Web Agent
- A Policy Server
- A user store
- A policy store
- An Administrative UI

You protect web-based resources by deploying at least one block.

The following diagram illustrates a simple deployment:



Each component has a specific role with resource protection.

Note: For more information about the primary purpose of each component, see Components.

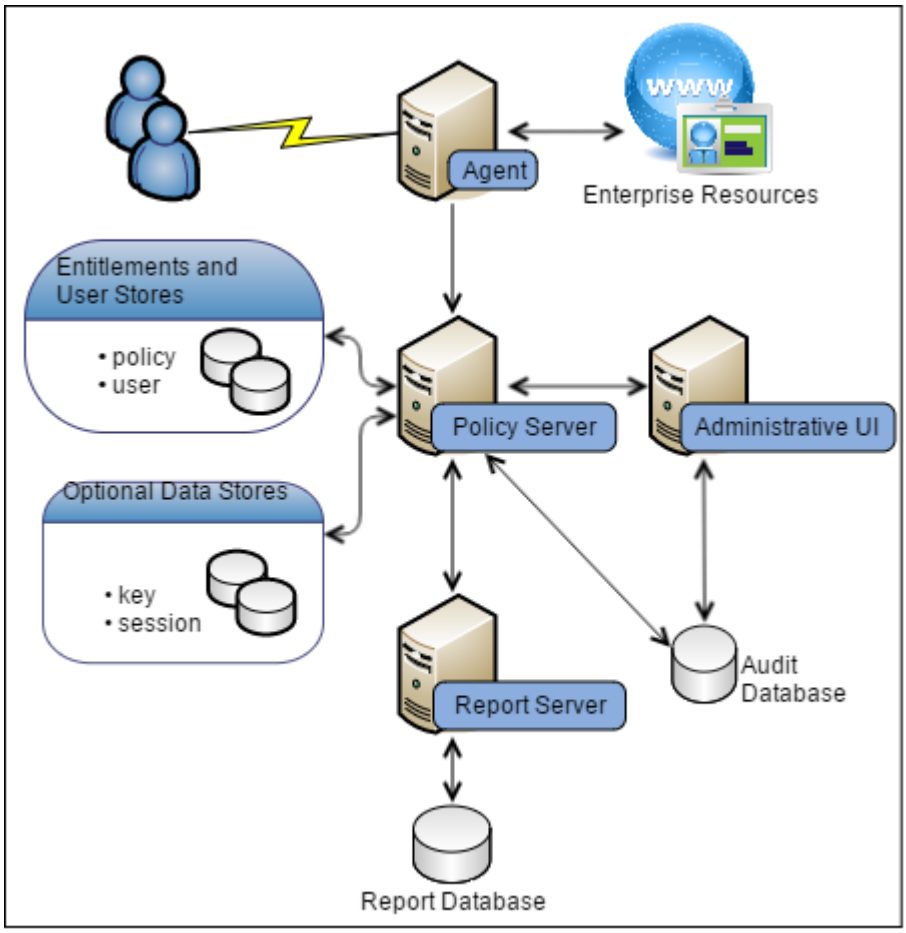
Simple Deployment with Optional Components

You can extend the functionality of a simple deployment through the use of optional CA Single Sign-On components. The decision to implement optional components is determined by the features your enterprise requires. For example:

- If you are planning to implement Federation–based functionality, your environment requires a certificate data store and a session store.
- If you are planning to create audit-based reports, your environment will require a Report Server and an audit database.

The following diagram illustrates the optional components and their required dependencies:

- A Report Server
- A report database
- An audit database
- A key store
- A session store
- A certificate data store



Each component has a specific role in resource protection.

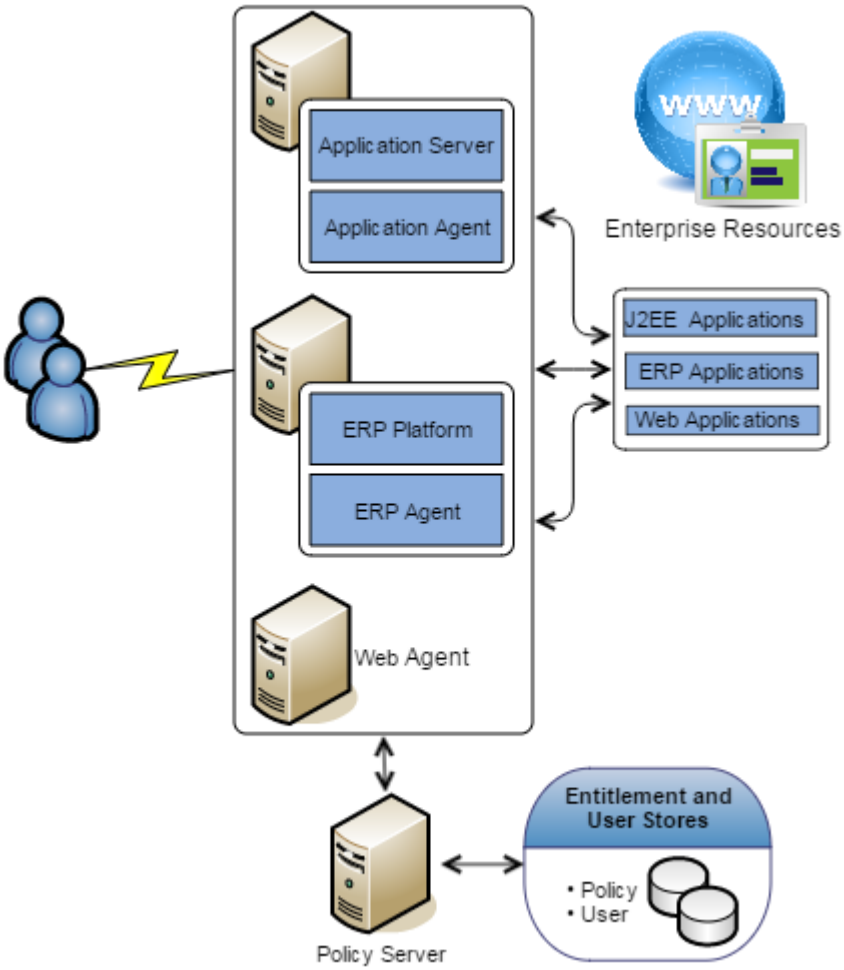
Note: For more information about the primary purpose of each component, see Components.

Simple Deployment with Optional Agents

You can extend the functionality of a simple deployment your environment to protect resources that do not reside on a Web Server. For example, if your environment hosts resources on an:

- Application server, you can implement Application Server Agents to protect them.
- ERP system, you can implement ERP Agents to protect them.

The following diagram illustrates optional Agents:



Each component has a specific role with resource protection.

Note: For more information about primary purpose of each component, see Components.

Multiple Components for Operational Continuity

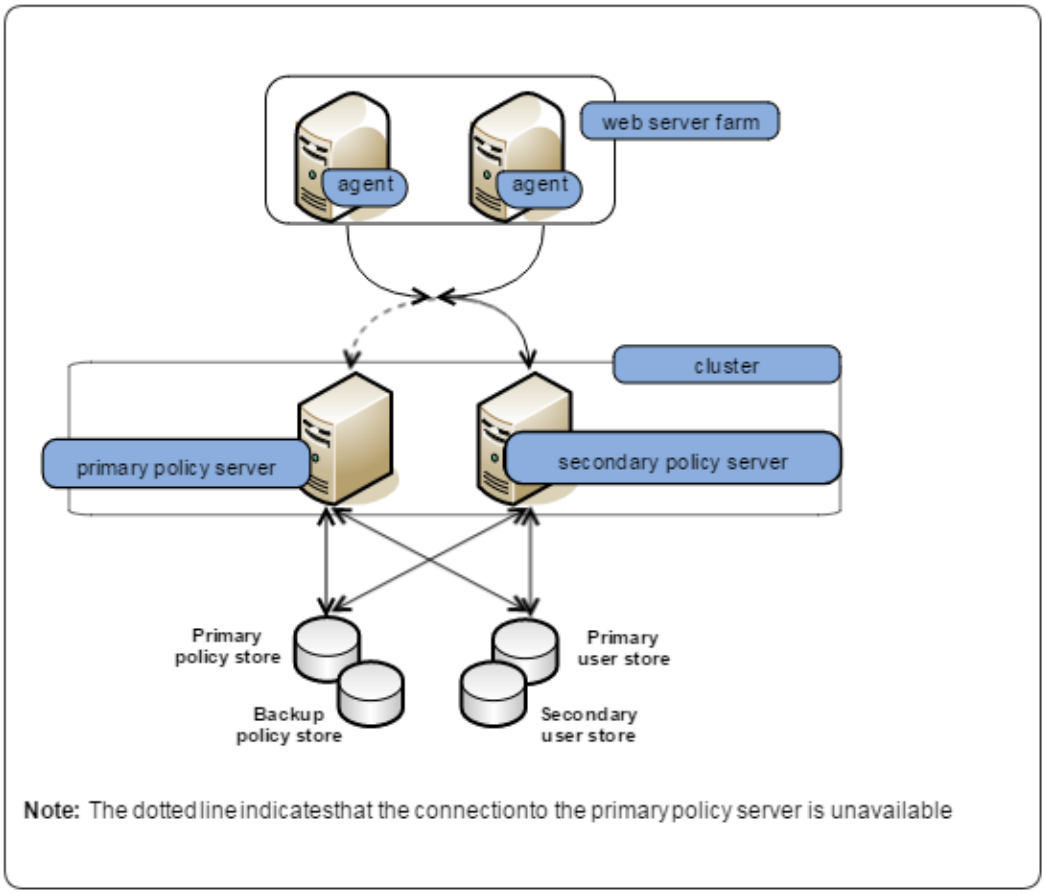
The following use cases show how you can implement multiple blocks of components to build redundancy and failover into the environment using the following methods:

- CA Single Sign-on round robin load balancing
- A hardware load balancer

Multiple Components for Operational Continuity Using CA Single Sign-On Load Balancing

You can implement multiple blocks of components to build redundancy and failover into the environment using CA Single Sign-On round robin load balancing. This use case builds on a simple deployment to explain how you can begin thinking about operational continuity. The following diagram illustrates:

- Multiple Agent instances intercepting user requests. As illustrated, each Agent is configured to initialize and communicate with a primary Policy Server and failover to the second Policy Server.
- A Policy Server cluster evaluating and enforcing access control policies. Load is dynamically distributed between each Policy Server in the cluster.
- Multiple user store connections. Each Policy Server is configured to communicate with a primary user store. The primary user store connection is configured with a secondary user store connection. The Policy Servers load balance requests for user information across both connections. If the primary connection becomes unavailable, Policy Servers failover to the secondary connection.
- A single policy store instance. Each Policy Server connects to the same policy store for a common view of policy information. The primary policy store connection is configured with a secondary connection to which the Policy Servers can failover.



Each component has a specific role with resource protection.

Note: For more information about the primary purpose of each component, see Components. For more information about redundancy and high availability, see Redundancy and High Availability.

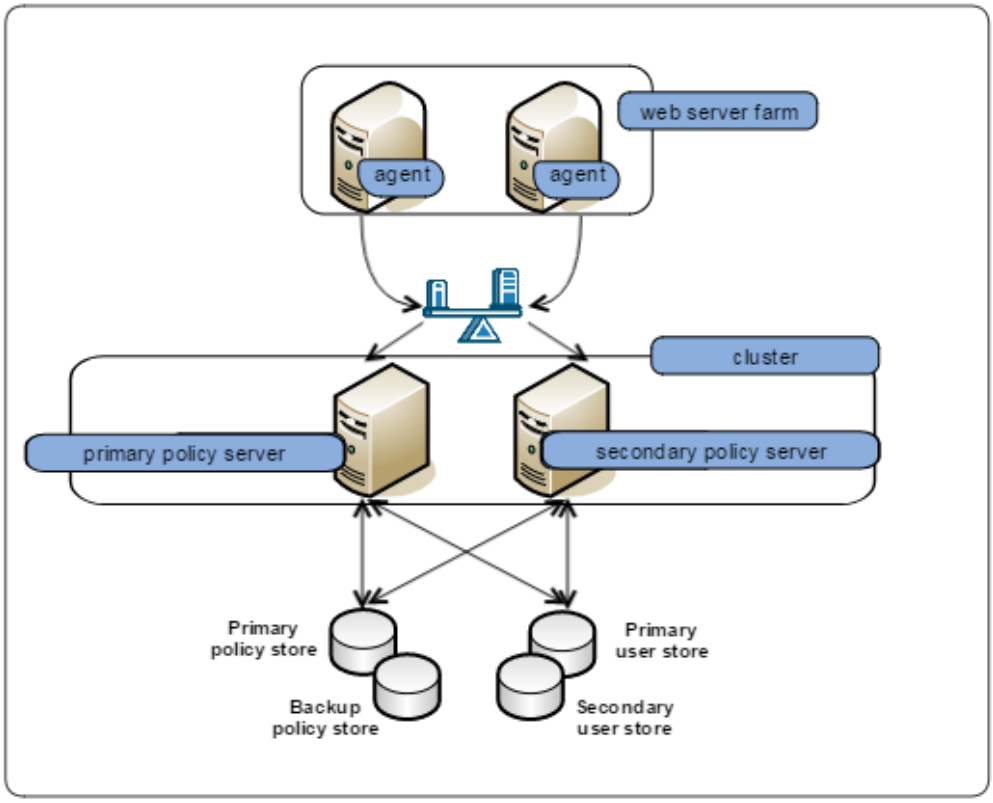
Multiple Components for Operational Continuity Using Hardware Load Balancing

You can implement multiple blocks of components to build redundancy and failover into the environment using hardware load balancing. This use case builds on a simple deployment to explain how you can begin thinking about operational continuity. The following diagram illustrates:

- Multiple Agent instances intercepting user requests. As illustrated, each Agent is configured to initialize and communicate with a primary Policy Server and failover to the second Policy Server.
- A hardware load balancer configured to expose multiple Policy Servers through a virtual IP address (VIP). The hardware load balancer dynamically distributes load between all Policy Servers associated with that VIP.
- Multiple Policy Servers evaluating and enforcing access control policies.

- Multiple user store connections. Each Policy Server is configured to communicate with a primary user store. The primary user store connection is configured with a secondary user store connection. The Policy Servers load balance requests for user information across both connections. If the primary connection becomes unavailable, Policy Servers failover to the secondary connection.

- A single policy store instance. Each Policy Server connects to the same policy store for a common view of policy information. The primary policy store connection is configured with a secondary connection to which the Policy Servers can failover.



Each component has a specific role with resource protection.

Note: For more information about the primary purpose of each component, see Components. For more information about redundancy and high availability, see Redundancy and High Availability.

Clustered Components for Scale

You can implement additional clusters to help performance levels remain high as you scale to extend throughput. This use case builds on the multiple components for operational continuity use case to explain how you can begin thinking about your architecture in terms of scale.

The initial deployment section of the diagram illustrates:

- A load balancer distributing user requests across multiple Agent clusters.
- Multiple Agent instances intercepting user requests for specific applications. Agents are configured to initialize and communicate with a primary Policy Server in the cluster. If enough Policy Servers in the cluster become unavailable, the Agents failover to another Policy Server cluster.

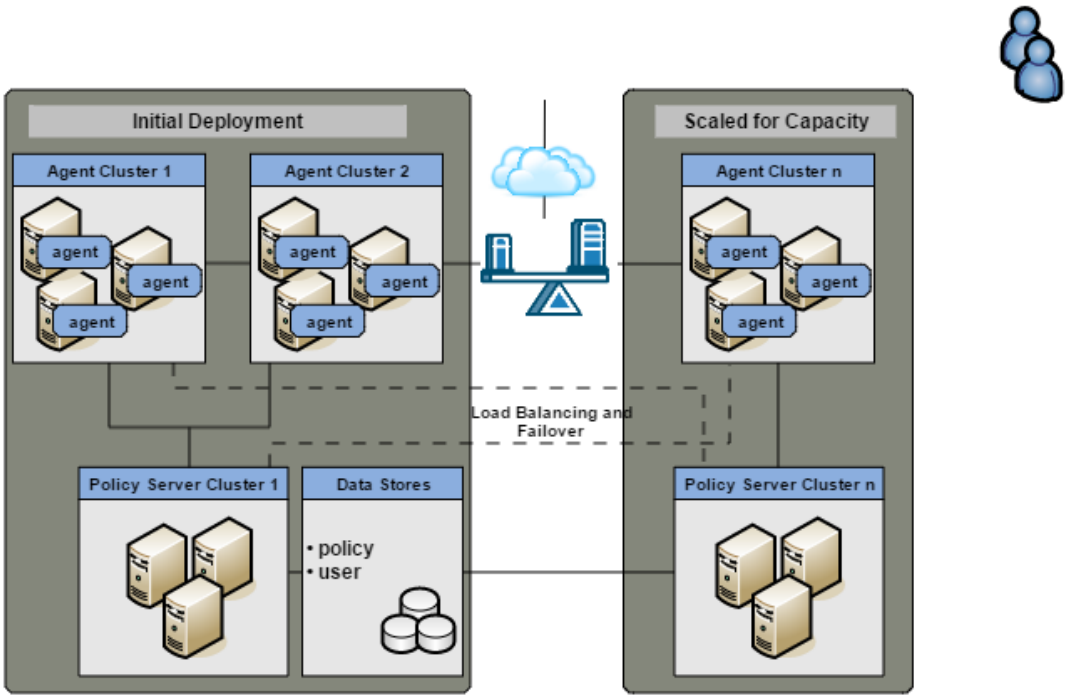
  Note: For more information about Agent and Policy Server redundancy and high availability, see Redundancy and High Availability.

- A Policy Server cluster evaluating and enforcing access control policies. Load is dynamically distributed between each Policy Server in the cluster.
- Multiple user store connections. Each Policy Server is configured to connect to a primary user store. The primary user store connection is configured with a secondary user store connection. The Policy Servers load balance requests for user information across both connections. If the primary connection becomes unavailable, Policy Servers failover to the secondary connection.

  Note: For more information about Policy Server and user store redundancy and high availability, see Redundancy and High Availability.

- A single policy store instance. Each Policy Server in the cluster connects to the same policy store for a common view of policy information. The primary policy store connection is configured with a secondary connection to which the Policy Servers can failover.

  Note: For more information about Policy Server and policy store redundancy, see Redundancy and High Availability.



Each component has a specific role with resource protection.

Note: For more information about the primary purpose of each component, see CA Single Sign-On Components.

The Scaled for Capacity section of the diagram details another component block and illustrates:

- A load balancer distributing requests to the new Agent cluster.
- Multiple Agent instances intercepting user requests. In addition to their connections to Policy Servers in Cluster n, each Agent can also be configured to failover to any Policy Server in the environment. As illustrated by the dotted line, the Agents in Agent Cluster n are configured to failover to the Policy Severs in Policy Server Cluster 1.
- A Policy Server cluster evaluating and enforcing access control policies. As illustrated by the dotted line, each Policy Server cluster is configured with a failover threshold. When the number of available Policy Servers falls below the specified threshold, all requests that the failed Policy Server would otherwise service are forwarded to another cluster.

Redundancy and High Availability

You configure redundancy and high availability between logical blocks of CA Single Sign-On components to maintain system availability and performance.

Agent to Policy Server Communication

When you configure a CA Single Sign-On Agent, a Host configuration file (named SmHost.conf by default), is created on the host server. The Agent uses the connection information in this Host configuration file to create an initial connection with a Policy Server.

After the initial connection is established, the Agent obtains subsequent Policy Server connection information from the Host Configuration Object (HCO) on the Policy Server.

You can configure the HCO to include multiple Policy Servers and specify the method the Agent uses to distribute requests among multiple Policy Servers.

A CA Single Sign-On Agent can distribute requests among multiple Policy Servers in the following ways:

- Failover
- Round–robin load balancing
- Round–robin load balancing over one or more clusters of Policy Servers

Alternatively, you can configure the HCO to include a single virtual IP address configured on a hardware load balancer to expose multiple Policy Servers. In this case, the load balancer is responsible for failover and load balancing, rather than the Agent software.

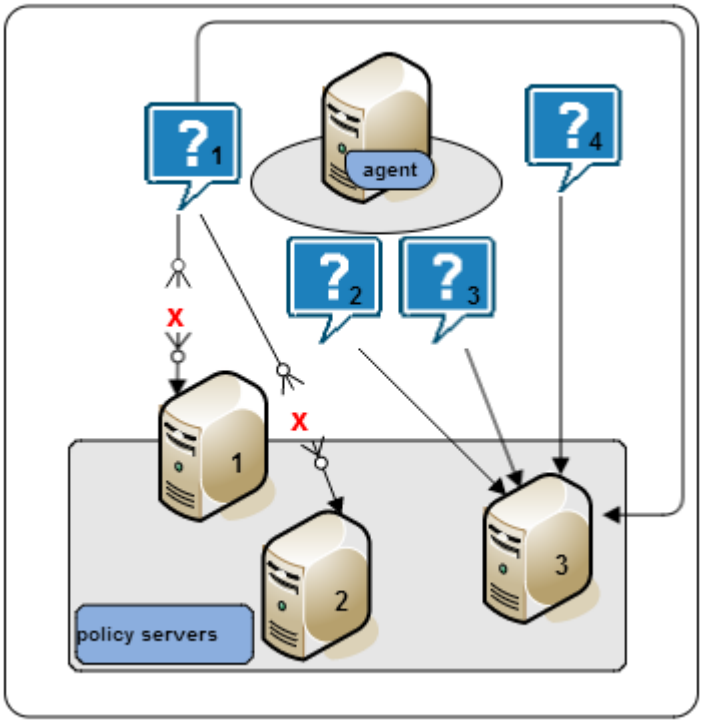
Failover

Failover is the default HCO setting. In failover mode, a CA Single Sign-On Agent delivers all requests to the first Policy Server that the HCO lists and proceeds as follows:

1. If the first Policy Server does not respond, the Agent deems it unavailable and redirects the request, and all subsequent requests, to the next Policy Server that the HCO lists.
2. If the first two Policy Servers do not respond, the Agent deems both of them unavailable and redirects the request, and all subsequent requests, to the next Policy Server that the HCO lists.

If an unresponsive Policy Server recovers, which the Agent determines through periodic polling, the Policy Server is automatically returned to its original place in the HCO list and begins receiving all Agent requests.

The following diagram illustrates the Agent failover process:



Round Robin Load Balancing

Round robin load balancing is an optional HCO setting. Round robin load balancing distributes requests evenly over a set of Policy Servers, which:

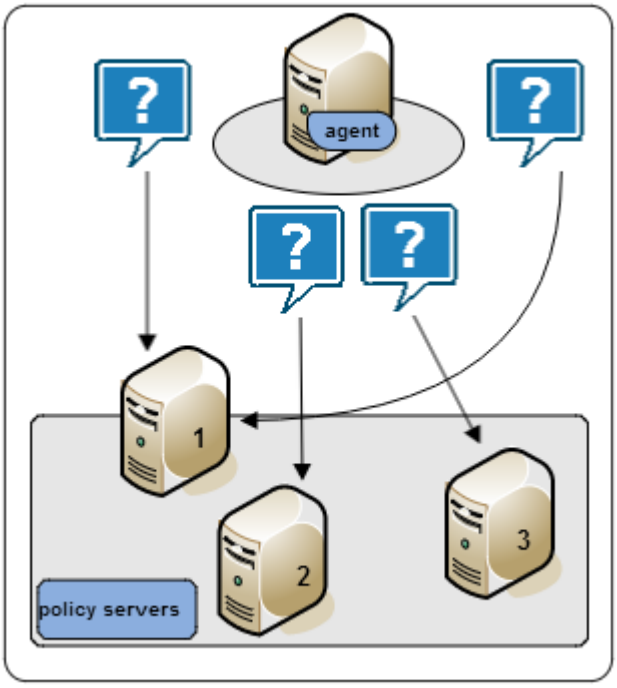
- Results in more efficient user authentication and authorization
- Prevents a single Policy Server from becoming overloaded with Agent requests

In round robin mode, an Agent distributes requests across all Policy Servers that the HCO lists. An Agent:

1. Sends a request to the first Policy Server that the HCO lists.
2. Sends a request to the second Policy Server that the HCO lists.
3. Sends a request to the third Policy Server that the HCO lists.
4. Continues sending requests in this way, until the Agent has sent requests to all available Policy Servers. After sending requests to all available Policy Servers, the Agent returns to the first Policy Server and the cycle begins again.

If a Policy Server does not respond, the Agent redirects the request to the next Policy Server that the HCO lists. If the unresponsive Policy Server recovers, which the Agent determines through periodic polling, the Policy Server is automatically restored to its original place in the HCO list.

The following diagram illustrates the round robin process:



Policy Server Clusters

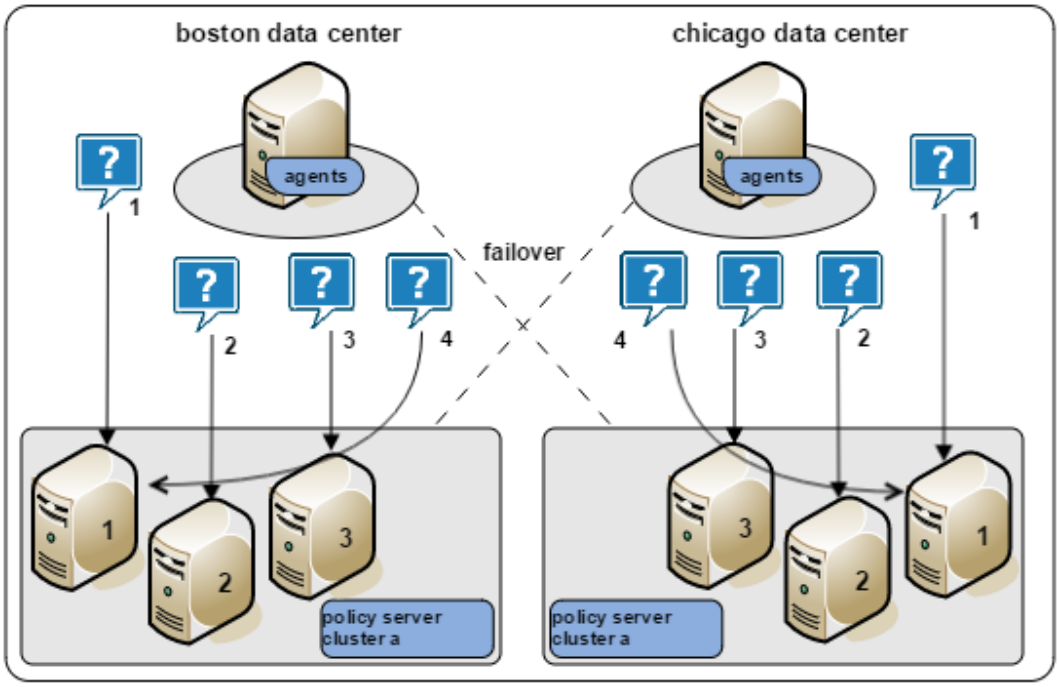
Round robin load balancing evenly distributes CA Single Sign-On Agent requests to all Policy Servers that the HCO lists. Although an efficient method to improve system availability and response times, consider that:

- Round robin load balancing is not the most efficient distribution method in a heterogeneous environment where computing capacity can differ. Each Policy Server receives the same number of requests, regardless of capacity.
- Round robin load balancing to Policy Servers that are located in different geographical locations can degrade performance. Sending Agent requests to Policy Servers outside a certain locale can result in increased network communication overhead and network congestion.

A Policy Server cluster is a group of Policy Servers to which Agents can distribute requests. Policy Server clusters provide the following benefits over round robin load balancing:

- A cluster can be configured to include Policy Servers only in a specific data center. Grouping Agents with distinct Policy Server clusters avoids the network overhead involved with load balancing across geographically separate regions. Network overhead is only incurred if Agents failover to another Policy Server cluster.
- A cluster can failover to another cluster based on a Policy Server failover threshold.
- Agents dynamically distribute requests across all Policy Servers based on response time, instead of distributing requests evenly.

The following diagram illustrates two Policy Server clusters. Each cluster is geographical separated to avoid the network overhead that can be associated with round robin load balancing.



Policy Server to User Store Communication

The Policy Server can distribute queries over multiple LDAP or ODBC user stores to enable the following:

- Failover
- Round robin load balancing

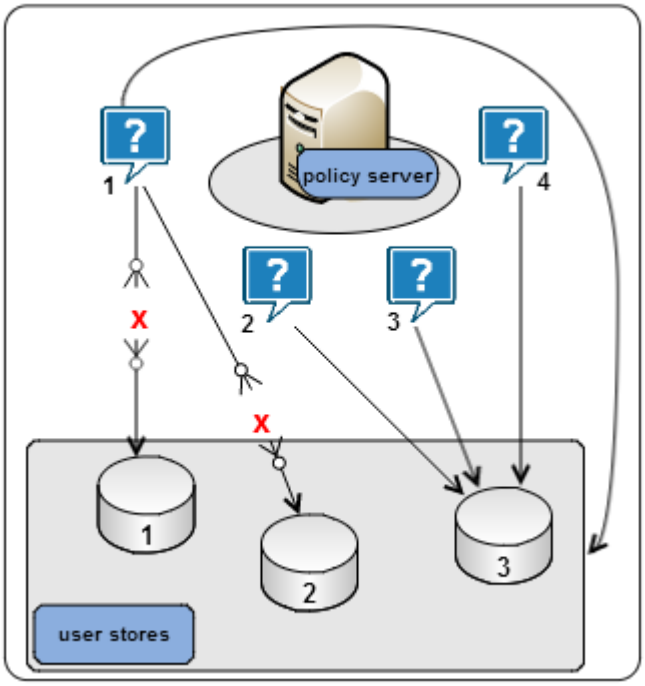
Failover

Failover is an optional setting in the CA Single Sign-On user store object. In failover mode, a Policy Server distributes all requests to the primary user store and proceeds as follows:

1. If the primary user store does not respond, the Policy Server deems it unavailable and redirects the request, and all subsequent requests, to the next user store that the CA Single Sign-On user store object lists.
2. If the first and second user stores do not respond, the Policy Server deems them both unavailable and redirects the request, and all subsequent requests to the next user store that the CA Single Sign-On user store object lists..

If an unresponsive user store recovers, the user store is automatically returned to its original place in the failover list and begins receiving all Policy Server requests.

The following diagram illustrates the user store failover process:



Round Robin Load Balancing

Round robin load balancing is an optional CA Single Sign-On user store object setting. Round robin load balancing distributes requests evenly over a set of user stores, which:

- Results in more efficient user store queries
- Prevents a single user store from becoming overloaded with Policy Server requests

Note: Consider the following:

- The Administrative UI does not include settings to configure round robin load balancing between ODBC user stores. However, the Policy Server installation includes:
  - The CA Single Sign-On Oracle Wire Protocol. This protocol supports load balancing over multiple Oracle stores. You can configure Oracle user store load balancing at the data source level.
  - The CA Single Sign-On SQL Server Wire Protocol, which you can use to configure SQL Server or SQL Server Cluster Enterprise. You can configure SQL Server user store load balancing at the database level.

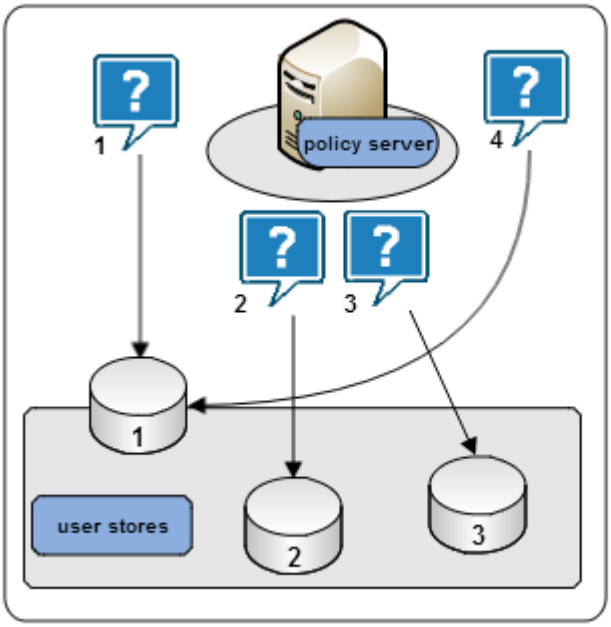
In round robin mode, a Policy Server distributes requests across all user stores that the CA Single Sign-On user store object lists. A Policy Server:

1. Sends a request to the first user store that the user store object lists.

2. Sends a request to the second user store that the user store object lists.
3. Sends a request to the third user store that the user store object lists.
4. Continues sending requests in this way, until the Policy Server has sent requests to all available user stores. After sending requests to all available user stores, the Policy Server returns to the first user store and the cycle begins again.

Configure load balancing with failover to add the benefit of redundancy in the event of a user store failure.

The following diagram illustrates the user store round robin process:



Policy Server to Policy Store Communication

All Policy Servers must connect to the same policy store for a common view of policy information. However, we recommend that the deployment includes multiple "hot" policy stores to which Policy Servers can failover.

The following are policy store failover scenarios:

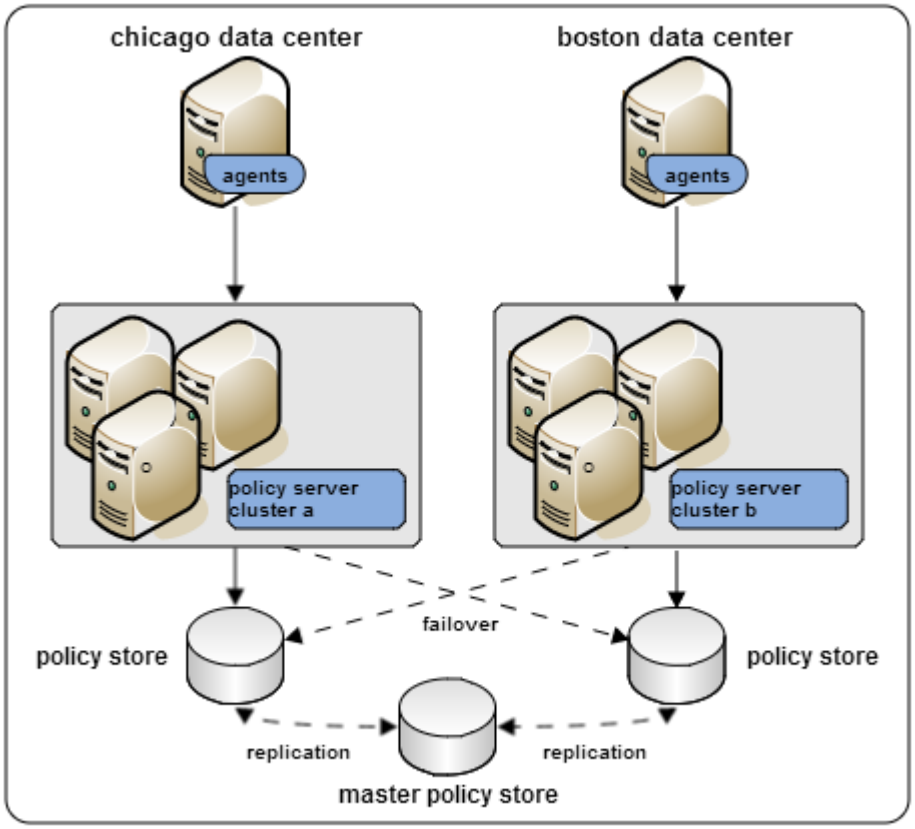
- A master policy store configured with replicated versions
- Multi–mastered policy stores

Master Policy Store

Deploying a master policy store with replicated versions is a way to achieve policy store redundancy. A single master policy store lets each Policy Server communicate with the closest replicated version. This method of communication:

- Improves the performance of geographically separated Policy Servers. Sending Policy Server requests to policy stores outside a certain locale can result in increased network communication overhead and network congestion.
- Allows for failover. If a primary policy store fails, Policy Servers failover to a secondary store.

The following diagram illustrates a single master policy store environment:



Multi–Mastered Policy Stores

Deploying LDAP directories using multi–master technology is a way to achieve policy store redundancy. A multi–master policy store lets each Policy Server communicate with the closest replicated version. This method of communication:

- Improves the performance of geographically separated Policy Servers. Sending Policy Server requests to policy stores outside a certain locale can result in increased network communication overhead and network congestion.
- Allows for failover. If a primary policy store fails, Policy Servers failover to a secondary store.

The following configuration is recommended when configuring an LDAP policy store in multi-master mode:

- A single master should be used for all administration.
- A single master should be used for key storage.
  This master does not need to be the same as the master used for Administration. However, we recommend that you use the same master store for both keys and administration. In this configuration, all key store nodes should point to the master rather than a replica.

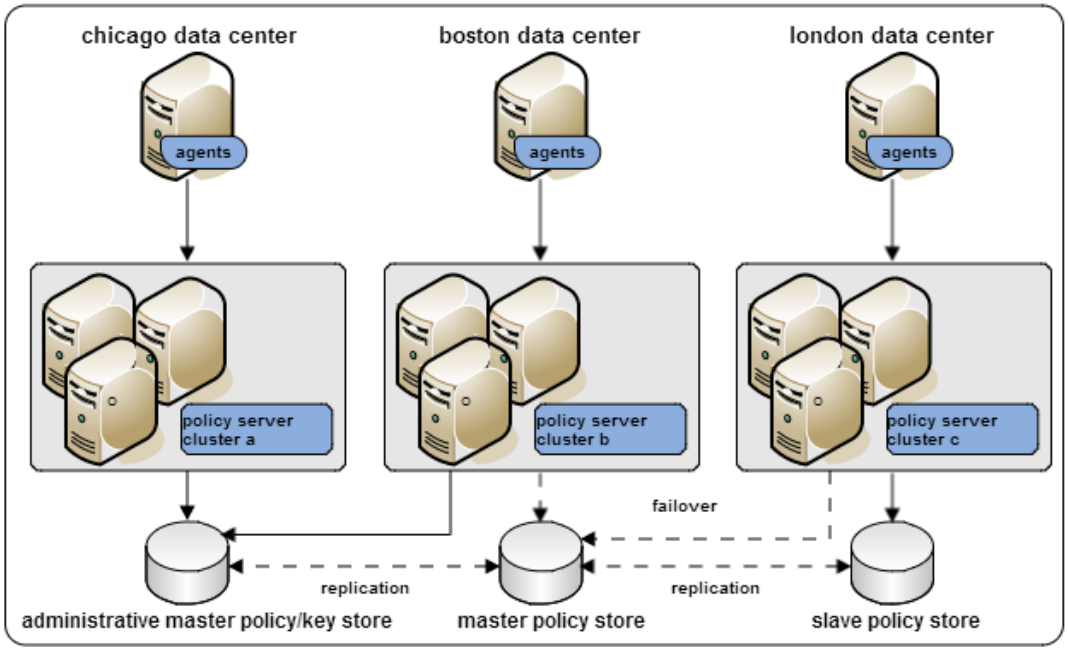
  Note: If you use a master for key storage other than the master for administration, then all key stores must use the same key store value. No key store should be configured to function as both a policy store and a key store.

- All other policy store masters should be set for failover mode.

Due to possible synchronization issues, other configurations may cause inconsistent results, such as policy store corruption or Agent keys that are out of sync.

Contact CA Single Sign-On Support for assistance with other configurations.

The following diagram illustrates a multi–master policy store environment:



Policy Server to Audit Store Communication

By default, each Policy Server stores its own audit information to a text file. This text file is known as the Policy Server log. You can configure a Policy Server to log audit data directly to a database.

CA Single Sign-On audit logs are typically used for audit and compliance purposes. Consider the following:

- Having all Policy Servers write to a centralized audit store, where all data can be queried at once, may be preferable. If you deploy a centralized audit store, we recommend a highly available deployment.

  Important! If you enable synchronous auditing, we recommend configuring failover to prevent an audit store outage from stopping all Policy Server authentications and authorizations. The Policy Server does not return the result of Agent authentication and authorization requests until the record is saved in the audit database. Users are not authenticated or authorized until the record is saved.

- If your deployment cannot permit Policy Servers to write to a centralized audit store, you can use the smauditimport utility to import individual Policy Server logs into a centralized audit store.

Policy Server to Session Store Communication

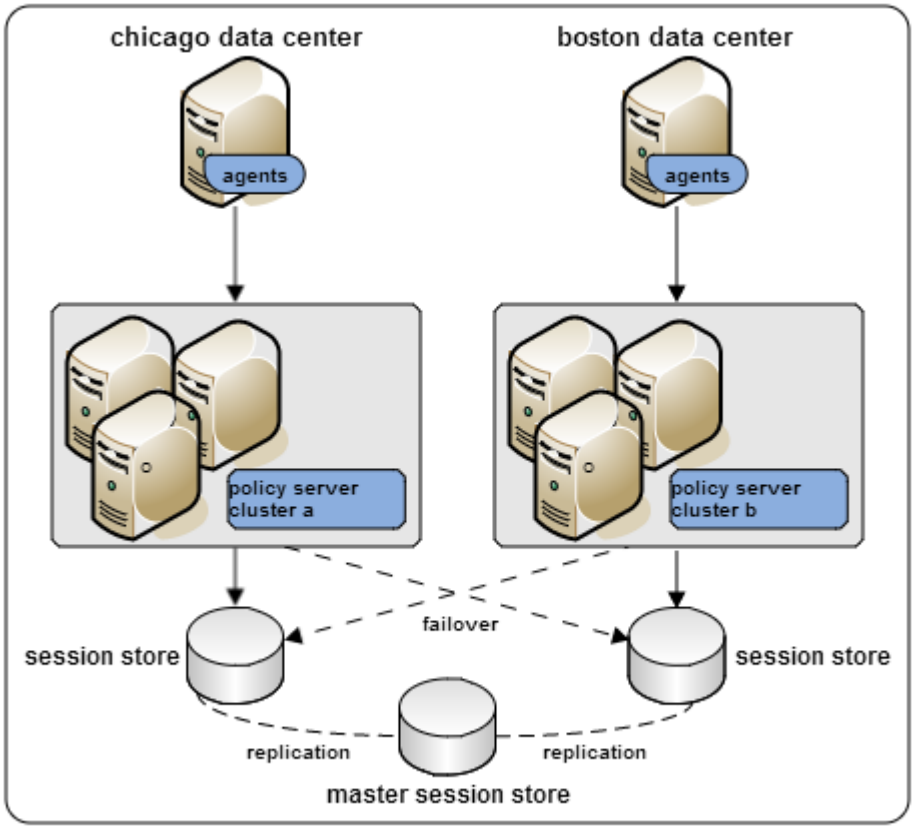
If you deploy a session store, all Policy Servers in the environment must use the same session store database.

Deploying a master session store is a way to achieve session store redundancy. A master session store lets each Policy Server communicate with the closest replicated version. This method of communication:

- Improves the performance of geographically separated Policy Servers. Sending Policy Server requests to a centralized session store outside a certain locale can result in increased network communication overhead and network congestion.

- Allows for failover. If a primary session store fails, Policy Servers failover to a secondary session store.

The following diagram illustrates all Policy Servers sharing a common view into a session store.



Architectural Considerations

The content in this section describes architectural information and use cases to consider when planning your CA Single Sign-On environment. Use the Table of Contents to access the content.

Plan an Implementation

This content describes how to plan a CA Single Sign-On implementation.

Policy Management Models

Policy management models let you define access permissions for web resources and their respective user populations. A policy management model establishes the following:

- What resource is protected.
- Who can access the resource.
- What type of access user populations have.

- What happens when CA Single Sign-On grants access to the resource.
- What happens when CA Single Sign-On denies access to a resource.

All functionality is available, regardless of which model you use. The primary difference between the models is the level of product knowledge required to configure each. The following Administrative UI objects represent the policy management models:

- Applications
- Policy domains and domain objects.
  The following core objects are required to configure an application object or domain policy:
  - A host configuration object
  - An agent configuration object
  - An agent object
  - A user directory object

Policy Management Using Application Objects

Application objects provide an intuitive method of defining a complete security policy for a web application, website, or web service. Applications associate resources with user roles to specify entitlement policies that determine what users can access what resources.

Note: An application object defines policy information that can otherwise be configured in a policy domain and its subobjects. That is, realms, rules, rule groups, responses, and policies. The following table summarizes this relationship.

Application Dialogs and Group Boxes	Equivalent Domain Component
General settings	Policy domain and the root location of the protected resources.
Components	Realm and the location of the resources within the application that share the same security requirements.
Resource	Rule and the required authentication or authorization actions.
Application Roles	User directory lookups.

Policy Management Using Policy Domains and Domain Objects

Before CA Single Sign-on r12.0, policy domains and domain objects (realms, rules, responses, policies, and so on) were the only way to protect resources. For Policy Server administrators already comfortable with them, policy domains and domain objects can still be used to configure security policies for resources.

A policy is comprised of the following individual objects:

- A domain
- At least one realm in the domain
- At least one rule or rule groups in the domain
- (Optional) One or more responses or response groups in the domain

A policy object binds these core objects to identify resources, user populations, and the required actions when access to a resource is granted or denied. As such, configuring a policy requires an understanding of each object.

Identify the Applications to Secure

Which applications are you planning to secure? How do they map to the access management models?

Begin thinking about the individual applications in the organization and the individual resources (URLs) within each application that require the same level of protection. We recommend identifying the following:

- Logical groupings of resources, often an individual application, that are associated with one or more user populations. These logical groupings map to either a policy domain or the resource filter of an EPM application. A policy domain or the resource filter of an EPM application represent the root location of the application.
- Sets of individual resources (URLs) within an application that have the same security (authentication and authorization) requirements. Sets of resources that share the same security requirements map to either a policy realm or an EPM application component.

Grouping resources in this way helps you map applications to the access management models.

When gathering information about each application, use a resource table similar to the following to help organization information:

Resource	Domain/Application Resource Filter	Realm/Component Resource Filter
Example: Corporate Portal	**Example:** Performance Management application	**Example:** Manager resources

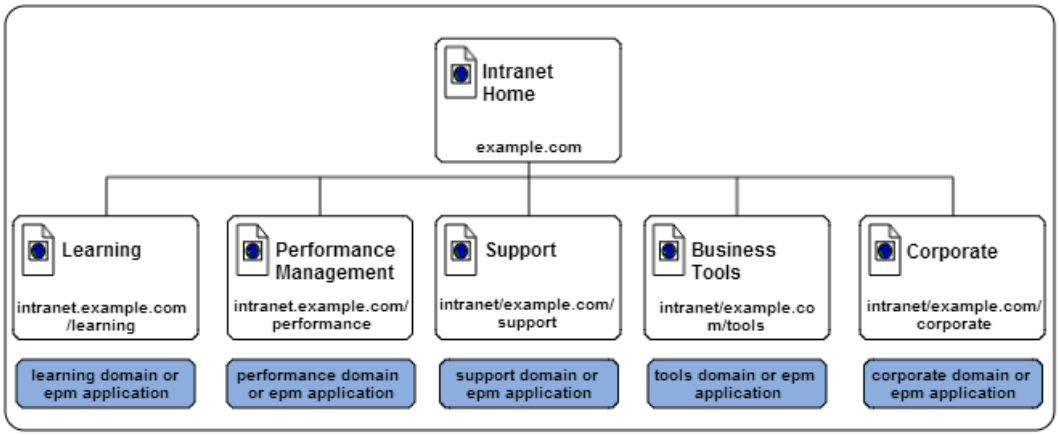
Note: Identifying the applications that require protection also aids in capacity planning.

Group Resources into Domains or EPM Applications

Defining a policy domain or an EPM application depends on identifying logical groups of resources, often an individual application, that are associated with one or more user populations. Grouping resources at this level helps you to identify the sets of individual resources (URLs) within an application that share the same security requirements.

A strategy for determining these requirements is to review a site map of the organization.

For example, example.com has a corporate intranet that the following site map represents:



In this example, the corporate portal is separated into the following logical groups of resources:

- Learning
- Performance Management
- Support
- Business Tools
- Corporate

The resource table for the corporate intranet looks like the following:

Resource	Domain/EPM Application Filter	Realm/Component Filter
Corporate Intranet	intranet.example.com	N/A
Learning	intranet.example.com/learning	N/A
Performance Management	intranet.example.com/performance	N/A
Support	intranet.example.com/support	N/A
Business Tools	intranet.example.com/tools	N/A
Corporate	intranet.example.com/corporate	N/A

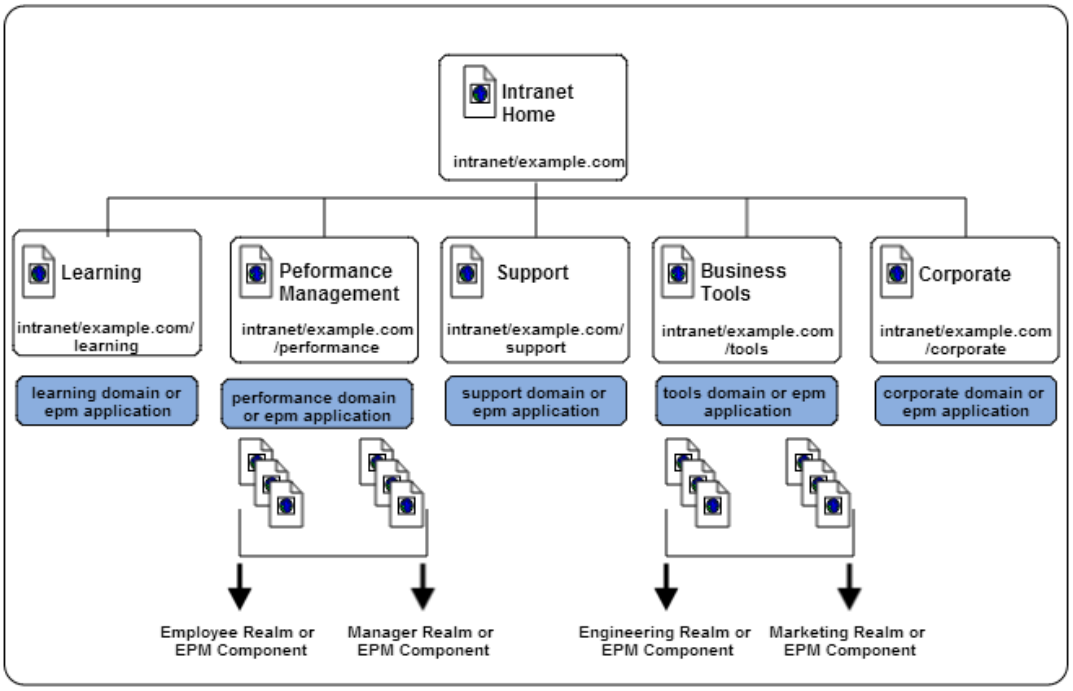
Group Resources into Realms or EPM Components

Defining a policy realm or an EPM component depends on identifying sets of individual resources (URLs) that share the same security or personalization requirements within a policy domain or EPM application. The contents of a realm or EPM component share the same authentication scheme. As a result, identifying these resources early in the process can help you determine the authentication schemes required to meet individual security requirements.

For example, although the Performance Management and Business Tools applications each let a specific user population access the root of the application, each application contains additional policy realms or EPM components to provide a level of security or personalization appropriate for the resource:

- The Performance Management application contains resources that only full–time employees can access and resources that only managers can access.
- The Business Tools application contains resources that only Research and Development employees can access and resources that only Marketing employees can access.

Note: Although not illustrated, policy rules and EPM resources are used to control specific Web Agent, authentication, and authorization events.



The resource table for the applications looks like the following:

Resource	Domain/EPM Application Filter	Realm/Component Filter
Corporate Intranet	intranet.example.com	N/A
Learning	intranet.example.com/learning	N/A
Performance Management	intranet.example.com/performance	/employee /manager
Support	intranet.example.com/support	N/A

Resource	Domain/EPM Application Filter	Realm/Component Filter
Business Tools	intranet.example.com/tools	/engineering /marketing
Corporate	intranet.example.com/corporate	N/A

Identify User Stores

CA Single Sign-On can authenticate and authorize users through one or more connections to existing user stores in your enterprise network. After you identify the applications to secure (see page 33), consider the following questions:

- Do the applications use a centralized user store or use separate user stores for authentication?
- If the applications use separate stores, does this project include a task to centralize the user identities into a single store?
- Do the applications use the same store to authenticate and authorize users? Or is a separate store or stores used for authorization?

Identifying the stores each application uses helps you to:

- Identify the user store connections a CA Single Sign-On Administrator must configure in a policy domain to protect the resource.
- Determine if your environment requires the directory mapping feature. By default, CA Single Sign-On assumes that users are authenticated and authorized against the same user store or stores. However, you can configure a policy domain to authenticate against one or more stores and authorize against others.

When gathering information about each application, use a table similar to the following to organize information:

User Store Name	User Store Type	Authentication?	Authorization?

Identify Authentication Methods

Multiple authentication methods are supported to meet the varying levels of protection your resources require:

Basic	MIT Kerberos
Forms-based user ID and password	Server-based, such as RADIUS and SafeWord

Hardware and software token-based, such as RSA® Ace /SecurID®	X.509 Certificate-based
Integrated Windows Authentication (IWA)	Custom Authentication schemes created using the CA Single Sign-On SDK
Information Card Authentication Schemes (ICAS), such as Microsoft Windows CardSpace	

After you identify the applications to secure (see page 33), in which we recommend identifying sets of resources (URLs) that share the same security requirements, consider the following questions:

- Are their authentication guidelines, regulations, or laws your organization is required to meet for specific types of resources?
- How sensitive and valuable is the information?
- What types of users are accessing this information?
- What type of security do these users expect?

Answering these types of questions helps you to

- Identify the authentication methods your environment requires
- Identify the authentication schemes a CA Single Sign-On Administrator must configure to protect a specific resource.

When gathering information about each resource, we recommend organizing your information by the applications you plan on securing. For example, the following table assumes that an application is grouped into individual domains and realms, as detailed in applications to secure (see page 33).

Resource	URL	Realm	Authentication Method

Identify Password Management Options

Do any security policies require your organization to manage user passwords? Do you anticipate managing user passwords in the future?

You can use password policies to enforce the password requirements of your enterprise. A password policy can validate the user's password against any of the following types of characteristics before accepting it:

- **Composition**

  Verifies the minimum or maximum length, the types of characters allowed, and if or how often any of those characters can be repeated in a password.

- **Age**

  Verifies the time limits for how long the same password can be used, how long a password can remain inactive before it must be changed, and how long or how often before an expired password can be reused. You can specify one of the following responses for users with expired passwords:

  - disable their accounts
  - force them to change their passwords

- **Attempts**

  Records the number of times the user has previously entered an incorrect password, and takes one of the following actions when that number is exceeded:

  - disables the account.
  - waits a specified time period before allowing either one login attempt or reenabling the account.

Password Policy Considerations

If you plan to implement password policies in your enterprise, consider the following:

- CA Single Sign-On requires read/write access to the user directory, including exclusive use of several attributes within that directory to store passwords and password–related information.
- Password policies can affect CA Single Sign-On performance because of the additional user directory searches required to validate passwords. Password policies that are configured to search only part of a user directory, instead of the entire directory, can also affect performance.
- If your user directory has a native password policy, this policy must be:

  - Less restrictive then the CA Single Sign-On password policy or
  - Disabled

  Otherwise the native password policy accepts or reject passwords without notifying CA Single Sign-On. Consequently, CA Single Sign-On cannot manage those passwords.

- By default, if a user enters incorrect information when changing a password, CA Single Sign-On returns a generic failure message. This message does not specify the failure reason. Create and enable the DisallowForceLogin registry key to change the default behavior and explicitly tell users why the change failed.

- If you use password policies on multiple Policy Servers, synchronize the system times of all servers. Synchronizing times helps to avoid the disabling of accounts or forcing password changes prematurely.

Identify Who Will Manage Your Web Agents

Web Agents connect to a Policy Server upon startup. The Policy Server contains an Agent Configuration Object (ACO), which directs the associated Web Agent to the location of its configuration parameters.

How your applications are deployed throughout your organization can help you determine the most efficient method of storing the configuration parameters for your CA Single Sign-On Web Agents. Consider the following questions:

Are most of your web applications deployed on a large server farm with the same security requirements?

1. Are most of your web applications managed by a centralized person or group?

2. Are most of your web applications deployed on separate web servers with different security requirements?

3. Are most of your web applications managed by different personnel in different departments or physical locations?

If you answered yes to questions one or two in the previous list, try the following configuration method:

- Central Configuration
  Manages the parameters for one or more agents from an agent configuration object (ACO) that resides in the Policy Server. With a central agent configuration, you can update the parameter settings of several agents at once. Generally, each distinct web application uses a separate ACO, whose settings are shared among all the agents that protect the web application. For example, if you have five agents protecting one accounting application, you can create one ACO with the settings that you want for the application. All five agents would use the parameter settings from the same ACO.
  For different applications, we recommend using separate agent configuration objects. For example, if you want to protect a human resources application with stricter security requirements, create a separate ACO for the human resources application.
  When an agent starts, it reads the AllowLocalConfig parameter values of its associated ACO. When the value is set to no, then the agent uses the parameter settings from the ACO (except the agent log and trace file settings). Agent log files and trace files can always be controlled locally, regardless of ACO settings.

  Note: We recommend using central agent configuration (wherever possible) because it simplifies agent configuration and maintenance.

If you answered yes to questions three or four in the previous list, try the following configuration method:

- Local Configuration
  Manages each Web Agent individually using a file installed on the web server itself. When a Web Agent starts, it reads the value of the AllowLocalConfig parameter of its associated Agent Configuration Object (ACO). If the value is set to yes, then the Web Agent uses the parameter settings from LocalConfig.conf file on the web server. The parameter settings from the LocalConfig.conf file override any settings stored in an ACO on the Policy Server.

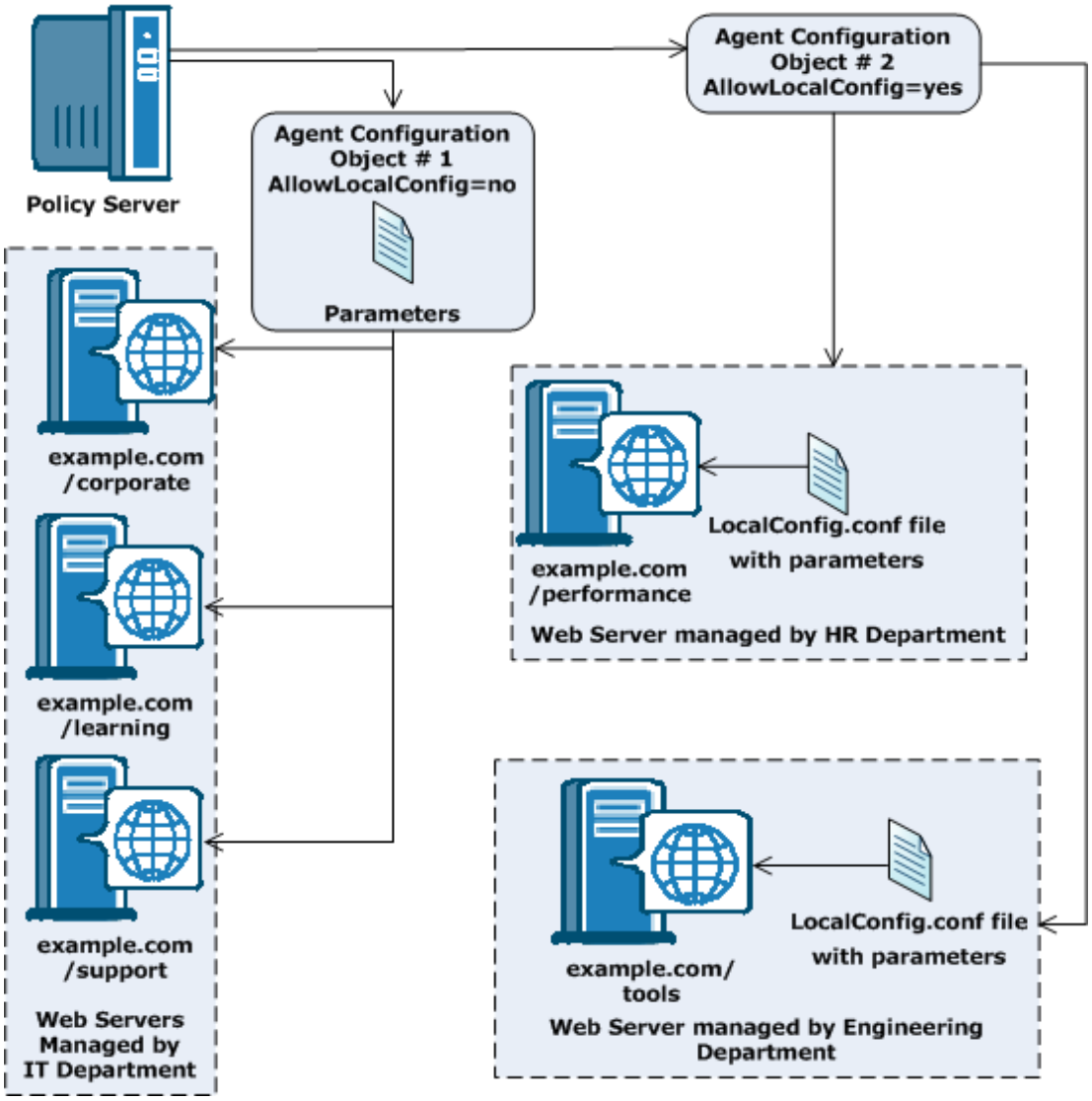
The following questions can help you identify other situations where local agent configuration better serves the needs of your enterprise:

- Will your enterprise deploy some of your Web Agents on reverse proxy servers?
  For example, you want to protect your internal resources with a large group of Web Agents, while implementing reverse-proxy servers in a few locations. You can use local configuration to manage the reverse-proxy Web Agents.

- Do you want to allow local web server administrators to change some Web Agent configuration settings but not others?
  For example, your organization uses CA Single Sign-On to manage and enforce security policies, but allows web server administrators in remote offices to customize their log on and log off pages. You can add individual parameters to the value of the AllowLocalConfig parameter of the ACO to allow the administrators to change only those settings for the customized pages but no others.

Central and Local Configurations Together

You can also use a combination of central and local configuration to meet your needs. For example, you can manage three similar web servers with central configuration, while managing the other two servers with local configuration. See the following illustration for an example:



Graphic shows that centerally configured agents receive Parameter Settings from a Single Object on the Policy Server while Locally-configured agents receive parameter settings from a local computer file

Identify Data Centers

Multiple factors, which are discussed later, can influence how you decide to implement CA Single Sign-On components across multiple data centers. Identifying the data centers and the purpose each is to serve in your CA Single Sign-On environment prepares you to make informed decisions when determining how to implement CA Single Sign-On components. Consider the following questions:

- How many data centers does your deployment include and where is each center located?
- If you have multiple data centers:
  - Will they all be active or are some only intended for disaster recover or backup?

- Will each protected application reside in a single data center or across multiple centers?
- Will you configure failover on the data center-level or across data centers?
- What is the bandwidth and throughput between the data centers?

When gathering information about each data center, use a resource table similar to the following to organize your results:

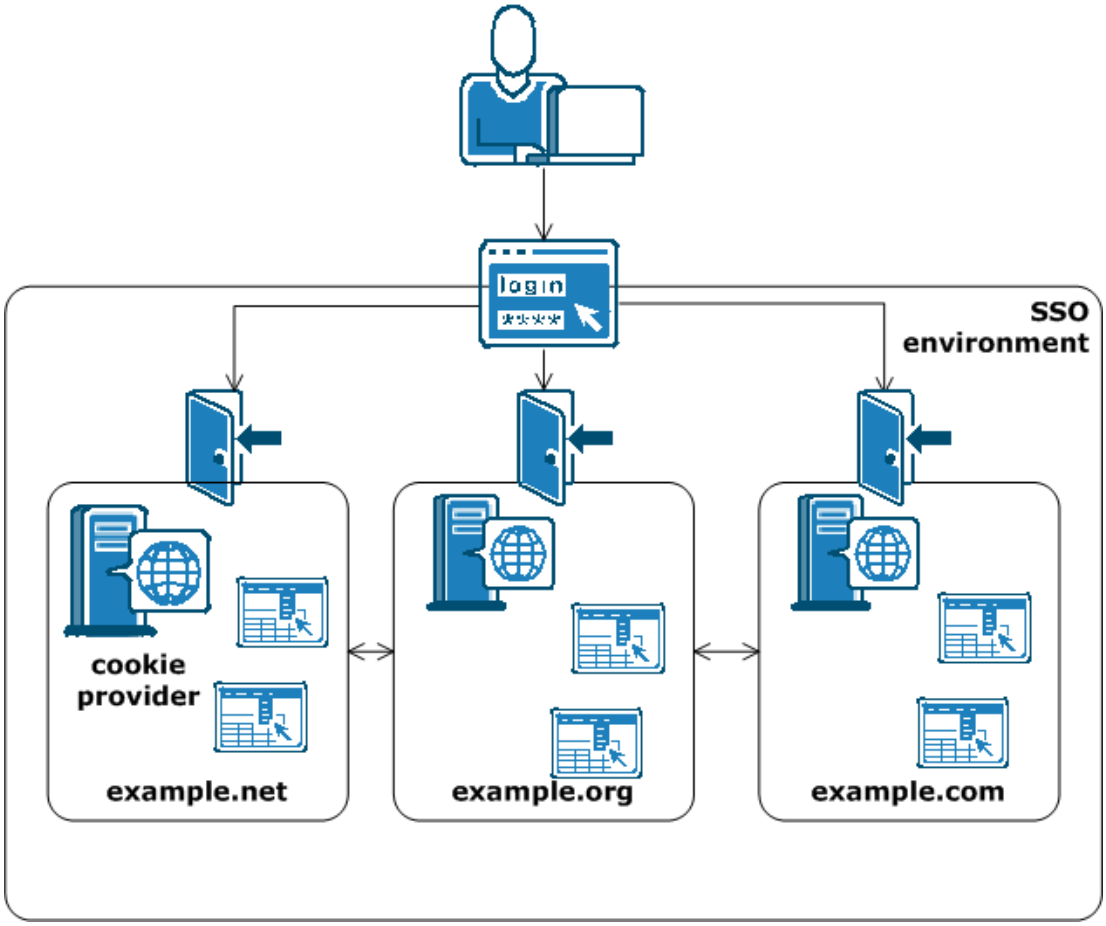
Data Center Name	Location	Purpose

Identify Resources to be Secured with Multiple Cookie Domains

Will the single-sign on environment in your enterprise extend across multiple cookie domains? CA Single Sign-On implements single sign-on across multiple cookie domains using a Web Agent configured as a *cookie provider*.

The cookie domain where the cookie provider resides is named the *cookie provider domain*. All the other Web Agents in the single sign-on environment point to the one cookie provider in the cookie provider domain.

The following illustration shows an example of an SSO environment using multiple cookie domains:

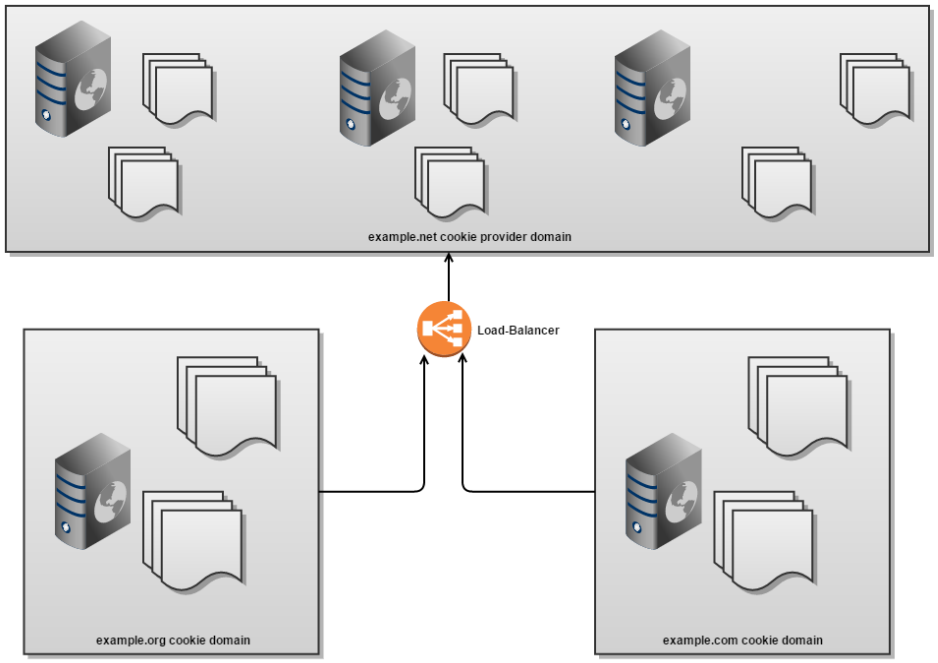


Graphic showing how users who authenticate to one domain can go to another without being re-challenged for their credentials

Load-balancing for SSO between Cookie Provider Domains and Other Cookie Domains

Will the Agents in your single-sign on environment use load-balancing?

All agents in an SSO environment must refer to a single cookie provider domain. Add a load-balancer between the web servers in your cookie provider domain and the other cookie domains in your SSO environment. The following illustration shows an example:



The Web Agent in the example.org cookie domain points and the Web Agent in the example.com cookie domain both point to the same cookie provider domain of example.net. A load-balancer distributes the traffic evenly between all the web servers in the example.net cookie provider domain.

Determine if Partnerships Require CA Single Sign-On Federation

Do existing or planned business-to-business (B2B) partnerships require your organization to share identity information securely with partners?

CA Single Sign-On Federation lets you extend CA Single Sign-On functionality to partner sites by enabling identity federation. CA Single Sign-On Federation offers two deployment options: legacy federation and partnership federation.

Federated transactions between partner organizations let your enterprise:

- Exchange user identity information between partners in a secure fashion.
- Establish a link between a user identity at a partner and a user identity in your company.
- Enable single sign–on across partner web sites in multiple domains.
- Handle different user session models between partner sites, such as single–logout across all partner web sites or separate sessions for each partner web site.
- Control access to resources based on user information that is received from a partner.
- Allow interoperability across heterogeneous environments.

CA Single Sign-On Federation lets your enterprise generate, consume, or generate and consume assertions. CA Single Sign-On Federation supports the following standards and protocols:

- SAML 1.0 (legacy federation only)
- SAML 1.1 and 2.0
- Microsoft ADFS/WS-Federation (legacy federation only)
- SAML browser artifact protocol
- SAML POST protocol
- WS-Federation Passive Requestor Profile protocol (legacy federation only)

If your organization plans on implementing federation, use a table similar to the following table to identify partners and the possible methods for enabling identity federation.

Partner	Standard	Protocol

Determine if Advanced Encryption Standards are Required

Does your organization require the use of Federal Information Processing Standard (FIPS) 140–2 compliant algorithms?

The CA Single Sign-On implementation of the Advanced Encryption Standard (AES) supports the FIPS 140–2 standard. FIPS is a US government computer security standard used to accredit cryptographic modules that meet the AES.

The Policy Server uses certified FIPS 140–2 compliant cryptographic libraries. These cryptographic libraries provide a FIPS mode of operation when a CA Single Sign-On environment only uses AES–compliant algorithms to encrypt sensitive data. A CA Single Sign-On environment can operate in one of the following FIPS modes of operation.

- FIPS–compatibility
- FIPS–migration
- FIPS–only

If you are implementing AES encryption through FIPS-only mode, consider the following:

- All third–party components, including directory servers, databases, and drivers must be configured to support FIPS–compliant algorithms.

Note: For more information about your vendors ability to support the FIPS 140–2 standard, see the vendor-specific documentation.

- If the environment uses X.509 Client Certificate authentication schemes, be sure that the user certificates are generated using only FIPS–compliant algorithms.
- If the Policy Servers are to connect to policy stores or user stores using SSL, be sure that the Policy Servers and directory stores use certificates that are FIPS–compliant.
- All Web Agents that ship with CA Single Sign-On r12.x are FIPS–compliant. To determine if other agents are FIPS–compliant, see the agent–specific documentation.

Important! An environment that is running in FIPS–only mode cannot operate with and is not backward compatible to earlier versions of CA Single Sign-On. This requirement includes all agents, custom software using older versions of the Agent API, and custom software using PM APIs or any other API that the Policy Server exposes. Re–link all such software with the current versions of the respective SDKs to achieve the required support for FIPS–only mode.

Determine if Virtualization is to be Used

Will CA Single Sign-On be implemented to a virtual environment?

Consider the following before implementing CA Single Sign-On to a virtual environment:

- Be sure to review the CA policy on virtualization (https://support.ca.com/irj/portal/phpsupcontent?techDocAccess=N&contentID=189545).
- Be sure to:
  - Understand the virtual environment and the performance overhead the host system can impose on applications.
  - Tune the virtual environment to eliminate as much as the performance overhead as possible.

    Note: For more information about performance tuning the virtual environment, see the vendor–specific documentation.
- Be sure to size the CPU, disk space, and memory available to the virtual environment. Use the system requirements section to determine how many components to deploy to the entire system.
- Be aware of issues associated with clock synchronization and multiple operating systems. Unsynchronized clocks can result in unexpected CA Single Sign-On behavior.

- When considering where to deploy components:
  - We recommend deploying Policy Servers to the virtual environment. We recommend that Policy Servers have their own Ethernet port. A dedicated port helps to prevent CA Single Sign-On from missing requests because it is competing with other virtual hosts for available bandwidth.
  - We recommend deploying Web Agents to virtualized web servers.
  - We recommend deploying all CA Single Sign-On data stores to physical hardware and operating systems. Directory servers and databases can become very resource dependent. If deployed to the virtualized environment, this dependency can result in performance degradation.

Determine how to Manage Policy Servers

Should individual business units be responsible for managing Policy Servers? Or can a single business unit manage all Policy Servers centrally?

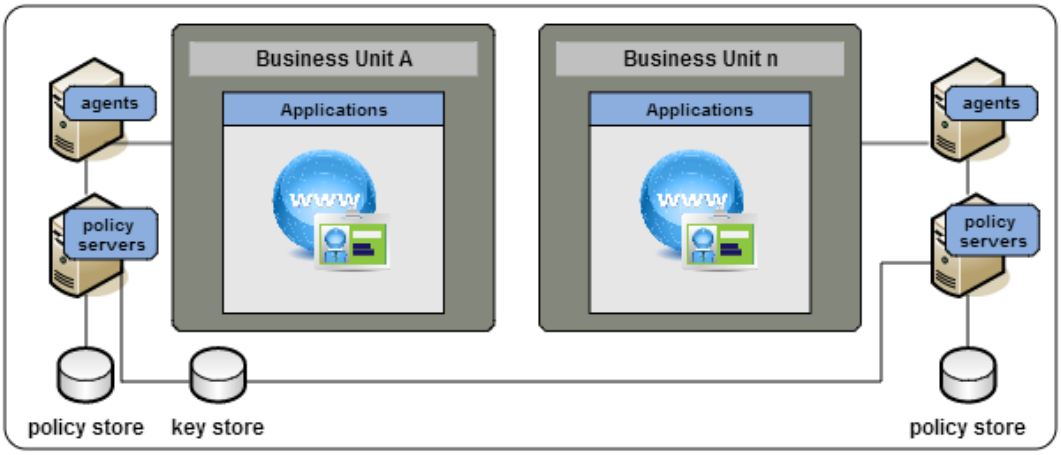
Local Policy Server Management

If individual business units manage Policy Servers and policy stores locally, consider that local Policy Server management:

- Lets each business unit manage their security requirements based on their individual needs.
- Can increase the complexity of the CA Single Sign-On infrastructure:
  - Local Policy Server management can result in more Policy Server and policy stores to manage and upgrade.
  - If single sign–on is a requirement, local Policy Server management results in additional CA Single Sign-On configuration. As illustrated, Policy Servers in both business units must share a key store to let all CA Single Sign-On Agents share the same keys.

    Note: The illustration details a shared key store to depict a single sign–on requirement. A shared key store is not the only way to implement single sign–on and additional requirements exist.
- Can make a consistent implementation and management of CA Single Sign-On core objects, policies, and EPM applications more challenging because CA Single Sign-On administrators are located in disparate business units.

The following illustration details two business units managing Policy Servers locally:



Central Policy Server Management

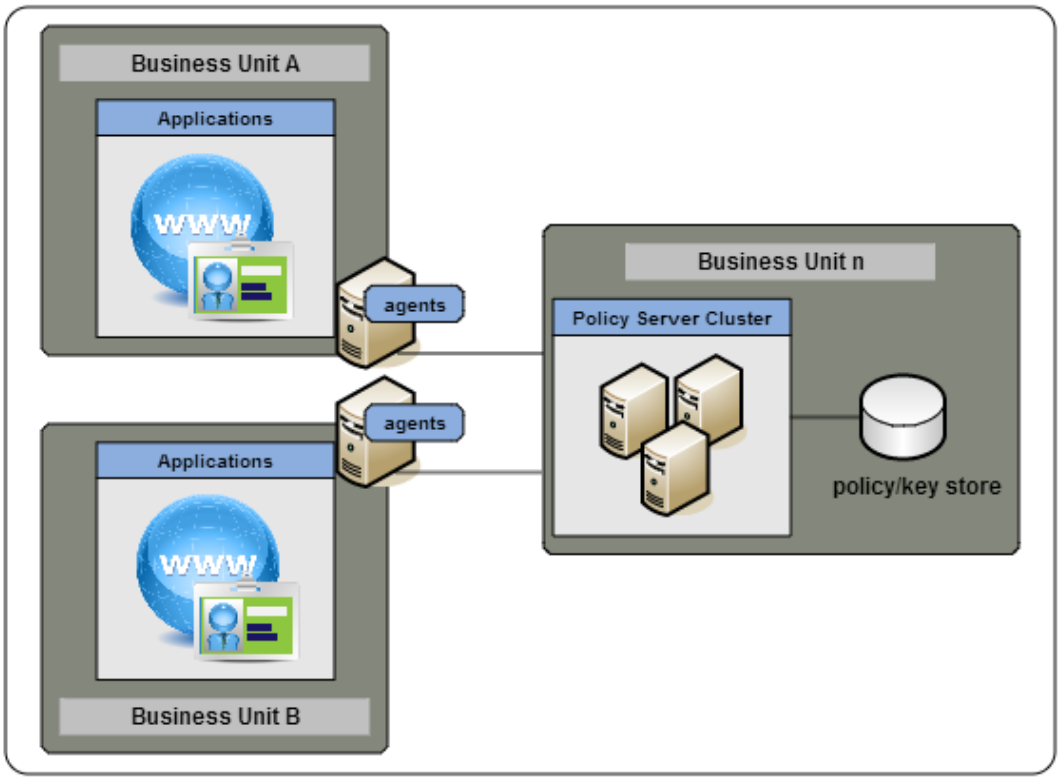
If a single business unit is to manage Policy Servers centrally, consider that central Policy Server management:

- Can facilitate a consistent implementation of CA Single Sign-On core objects, policies, and EPM applications because all CA Single Sign-On administrators are located in the same business unit.
- Can make the management of these objects easier because all CA Single Sign-On administrators are located in the same business unit.

  Note: As illustrated, individual business units can continue to manage the CA Single Sign-On Agents protecting their applications.

- Can simplify the CA Single Sign-On infrastructure. Central management can result in fewer Policy Server and policy stores to manage and upgrade.
- Lets administrators monitor CA Single Sign-On performance centrally.

The following illustration details a single business unit managing all Policy Servers:



Determine how to Manage Web Agents

If you have several Web Agents which will all be configured identically, then using an Agent Configuration object on the Policy Server will make managing your Web Agents easier. A single Agent configuration object can be shared among an unlimited number of Web Agents. Configuration changes made on the Policy Server are automatically applied to any Web Agents which use the configuration object.

Plan a Web Services Security Implementation

This content describes how to plan a Web Services Security (WSS) implementation.

- Determine how to Manage Policy Servers (see page 57)
- Determine how to Manage CA Single Sign-on WSS Agents (see page 58)

Policy Management Models

CA Single Sign-On Web Services Security access management models let you define access permissions for applications and their respective user populations. An access management model establishes the following:

- What resource is protected.
- Who can access the resource.
- What type of access user populations have.
- What happens when CA Single Sign-On grants access to the resource.
- What happens when CA Single Sign-On denies access to a resource.

Almost all CA Single Sign-On Web Services Security functionality is available, regardless of which model you use. The primary difference between the models is the level of CA Single Sign-On knowledge required to configure each. The following Administrative UI objects represent the policy management models:

- Application objects
- Policy domains and policy objects

Note: The following CA Single Sign-On core objects are required to configure an application object or CA Single Sign-On domain policy:

- A host configuration object
- An agent configuration object
- An agent object
- A user directory object
  For more information about these objects, see the <*co*>.

Policy Management Using Application Objects

The recommended method for creating and managing new security policies for your CA Single Sign-On Web Services Security environment is to define application objects that represent one or more related web services and then generate the component and resource settings that define what to protect from associated WSDL files.

Note: Application objects do not support policy expressions using variable objects. Content-based authorization using variables must be implemented using policy domains and policies.

Policy Management Using Policy Domains and Policies

For Policy Server administrators already comfortable with CA SOA Security Manager or CA Single Sign-On, policy management using policy domains and domain objects (realms, rules, responses, policies, and so on) — can still be used to perform manual configuration of security policies for web service resources.

Domains and domain objects must also be used in the following situations:

- To modify policies created traditionally and migrated from a previous CA SOA Security Manager deployment.
- To implement content-based authorization using variables.

Identify the Web Services to Secure

Which web services are you planning to secure? How do they map to the CA Single Sign-On Web Services Security policy management methods?

Begin thinking about the individual web services in your organization and the sets of operations within each web service that require the same level of protection. We recommend identifying the following:

- Logical groupings of resources, often offered by individual web services, that are associated with one or more user populations.
- Sets of individual operations within a web service that have the same security (authentication and authorization) requirements.

Note: Identifying the web services that require protection also aids in capacity planning.

Identify User Stores

CA Single Sign-On Web Services Security can authenticate and authorize users through one or more connections to existing user stores in your enterprise network. After you identify the web services to secure, consider the following questions:

- Do the web services use a centralized user store or use separate user stores for authentication?
- If the web services use separate stores, does this project include a task to centralize the user identities into a single store?
- Do the web services use the same store to authenticate and authorize users? Or is a separate store or stores used for authorization?

Identifying the stores each web service uses helps you to:

- Identify the user store connections a CA Single Sign-On Administrator must configure in a CA Single Sign-On policy domain to protect the resource.
- Determine if your environment requires the CA Single Sign-On directory mapping feature. By default, CA Single Sign-On assumes that users are authenticated and authorized against the same user store or stores. However, you can configure a CA Single Sign-On policy domain to authenticate against one or more stores and authorize against others.

When gathering information about each web service, use a table similar to the following to organize information:

User Store Name	User Store Type	Authentication?	Authorization?

Identify Authentication Methods

CA Single Sign-On Web Services Security supports four authentication methods to meet the varying levels and types of protection your resources require:

- **XML Document Credential Collector**
  Validates XML messages using credentials gathered from the message itself by mapping fields within the document to fields within a user directory.
- **XML Digital Signature**
  Validates XML documents digitally signed with valid X.509 certificates.
- **WS-Security**
  Validates XML messages using credentials gathered from WS-Security headers in the SOAP envelope of an incoming message.
  CA Single Sign-On Web Services Security can produce and consume WS-Security tokens, enabling you to use the WS-Security authentication scheme to deploy a multiple-web service implementation across federated sites.
- **SAML Session Ticket**
  Validates XML messages using credentials obtained from CA Single Sign-On Web Services Security synchronized-sessioning SAML assertions (which contain an encrypted combination of a CA Single Sign-on session ticket and a CA Single Sign-on user public key) placed in the message HTTP header, SOAP envelope, or cookie.
  CA Single Sign-On Web Services Security can generate and consume SAML Session Ticket assertions. This enables you to use the SAML Session Ticket authentication scheme to deploy a multiple-web service implementation within a single Policy Server domain.

After you identify the web services to secure, in which we recommend identifying web service operations that share the same security requirements, consider the following questions:

- Are their authentication guidelines, regulations, or laws your organization is required to meet for specific types of resources?
- How sensitive and valuable is the information?
- What types of users are accessing this information?
- What type of security do these users expect?

Answering these types of questions helps you to

- Identify the authentication methods your environment requires
- Identify the authentication schemes a CA Single Sign-On Administrator must configure to protect a specific resource.

Identify Who Will Manage Your CA Single Sign-on WSS Agents

CA Single Sign-on WSS Agents connect to a Policy Server upon startup. The Policy Server contains an Agent Configuration Object (ACO), which directs the associated CA Single Sign-on WSS Agent to the location of its configuration parameters.

How your web services are deployed throughout your organization can help you determine the most efficient method of storing the configuration parameters for your CA Single Sign-on WSS Agents. Consider the following questions:

Are most of your web services deployed on a large server farm with the same security requirements?

1. Are most of your web services managed by a centralized person or group?
2. Are most of your web services deployed on separate servers with different security requirements?
3. Are most of your web services managed by different personnel in different departments or physical locations?

CA Single Sign-On Web Services Security offers the following configuration methods:

- **Central Configuration**
  If you answered yes to questions one or two in the previous list, try central configuration in which one or more CA Single Sign-on WSS Agents are managed from an Agent Configuration Object (ACO) that resides in the Policy Server. With central configuration, you can update the parameter settings of several CA Single Sign-on WSS Agents at once. Generally, each distinct web service uses a separate ACO, whose settings are shared among all the CA Single Sign-on WSS Agents that protect the web service. For example, if you have five CA Single Sign-on WSS Agents protecting one accounting web service, you can create one ACO with the settings for the web service. All five CA Single Sign-on WSS Agents would use the parameter settings from the same ACO.
  For different applications, we recommend using separate Agent Configuration Objects. For example, if you want to protect a human resources web service with stricter security

requirements, create a separate ACO for the human resources web service.
When a CA Single Sign-on WSS Agent starts, it reads the value of the AllowLocalConfig parameter of its associated ACO. If the value is set to no, then the CA Single Sign-on WSS Agent uses the parameter settings from the ACO.

Note: We recommend using central agent configuration (wherever possible) because it simplifies agent configuration and maintenance.

- **Local Configuration**
  If you answered yes to questions three or four in the previous list, try local configuration in which each CA Single Sign-on WSS Agent is managed individually using a file installed on the server itself. When a CA Single Sign-on WSS Agent starts, it reads the value of the AllowLocalConfig parameter of its associated Agent Configuration Object (ACO). If the value is set to yes, then the CA Single Sign-on WSS Agent uses the parameter settings from LocalConfig.conf file on the application or web server. The parameter settings from the LocalConfig.conf file override any settings stored in an ACO on the Policy Server.If you answered yes to questions three or four in the previous list, try the following configuration method:

You can also use a combination of central and local configuration to meet your needs. For example, you can manage three similar web servers with central configuration, while managing the other two servers with local configuration.

The following questions can help you identify other situations where local agent configuration better serves the needs of your enterprise:

- Will your enterprise deploy custom CA Single Sign-on WSS Agents on XML gateways?
  For example, you want to protect your internal resources with a large group of CA Single Sign-on WSS Agents, while implementing XML gateways in a few locations. You can use local configuration to manage the custom CA Single Sign-on WSS Agents on XML gateways.

- Do you want to allow local server administrators to change some CA Single Sign-on WSS Agent configuration settings but not others?
  For example, your organization uses CA Single Sign-On to manage and enforce security policies, but allows application and web server administrators in remote offices to customize their log on and log off pages. You can add individual parameters to the value of the AllowLocalConfig parameter of the ACO to allow the administrators to change only those settings for the customized pages but no others.

Identify Data Centers

Multiple factors, which are discussed later, can influence how you decide to implement CA Single Sign-On components across multiple data centers. Identifying the data centers and the purpose each is to serve in your CA Single Sign-On environment prepares you to make informed decisions when determining how to implement CA Single Sign-On components. Consider the following questions:

- How many data centers does your deployment include and where is each center located?

- If you have multiple data centers:

- Will they all be active or are some only intended for disaster recover or backup?
- Will each protected application reside in a single data center or across multiple centers?
- Will you configure failover on the data center-level or across data centers?
- What is the bandwidth and throughput between the data centers?

When gathering information about each data center, use a resource table similar to the following to organize your results:

Data Center Name	Location	Purpose

Determine if Advanced Encryption Standards are Required

Does your organization require the use of Federal Information Processing Standard (FIPS) 140–2 compliant algorithms?

The CA Single Sign-On implementation of the Advanced Encryption Standard (AES) supports the FIPS 140–2 standard. FIPS is a US government computer security standard used to accredit cryptographic modules that meet the AES.

The Policy Server uses certified FIPS 140–2 compliant cryptographic libraries. These cryptographic libraries provide a FIPS mode of operation when a CA Single Sign-On environment only uses AES–compliant algorithms to encrypt sensitive data. A CA Single Sign-On environment can operate in one of the following FIPS modes of operation.

- FIPS–compatibility
- FIPS–only

If you are implementing AES encryption through FIPS-only mode, consider the following:

- All third–party components, including directory servers, databases, and drivers must be configured to support FIPS–compliant algorithms.

  Note: For more information about your vendors ability to support the FIPS 140–2 standard, see the vendor-specific documentation.

- If the environment uses X.509 Client Certificate authentication schemes, be sure that the user certificates are generated using only FIPS–compliant algorithms.

- If the Policy Servers are to connect to policy stores or user stores using SSL, be sure that the Policy Servers and directory stores use certificates that are FIPS–compliant.
- All CA Single Sign-on WSS Agents that ship with CA Single Sign-On are FIPS–compliant. To determine if other agents are FIPS–compliant, see the agent–specific documentation.

Important! An environment that is running in FIPS–only mode cannot operate with and is not backward compatible to earlier versions of CA Single Sign-On. This requirement includes all agents, custom software using older versions of the CA Single Sign-On Web Services Security SDK. Re–link all such software with the current versions of the SDK to achieve the required support for FIPS–only mode.

Determine if Virtualization is to be Used

Will CA Single Sign-On be implemented to a virtual environment?

Consider the following before implementing CA Single Sign-On to a virtual environment:

- Be sure to review the CA policy on virtualization (https://support.ca.com/irj/portal/phpsupcontent?techDocAccess=N&contentID=189545).
- Be sure to:
  - Understand the virtual environment and the performance overhead the host system can impose on applications.
  - Tune the virtual environment to eliminate as much as the performance overhead as possible.

    Note: For more information about performance tuning the virtual environment, see the vendor–specific documentation.
- Be sure to size the CPU, disk space, and memory available to the virtual environment. Use the system requirements section to determine how many components to deploy to the entire system.
- Be aware of issues associated with clock synchronization and multiple operating systems. Unsynchronized clocks can result in unexpected CA Single Sign-On behavior.
- When considering where to deploy components:
  - We recommend deploying Policy Servers to the virtual environment. We recommend that Policy Servers have their own Ethernet port. A dedicated port helps to prevent CA Single Sign-On from missing requests because it is competing with other virtual hosts for available bandwidth.

- We recommend deploying Web Agents to virtualized web servers.
- We recommend deploying all CA Single Sign-On data stores to physical hardware and operating systems. Directory servers and databases can become very resource dependent. If deployed to the virtualized environment, this dependency can result in performance degradation.

Determine how to Manage Policy Servers

Should individual business units be responsible for managing Policy Servers? Or can a single business unit manage all Policy Servers centrally?

- **Local Policy Server Management**

If individual business units manage Policy Servers and policy stores locally, consider that local Policy Server management:

- Lets each business unit manage their security requirements based on their individual needs.
- Can increase the complexity of the CA Single Sign-On infrastructure; local Policy Server management can result in more Policy Server and policy stores to manage and upgrade.
- Can make a consistent implementation and management of CA Single Sign-On core objects, policies, and application objects more challenging because CA Single Sign-On administrators are located in disparate business units.
- **Centralized Policy Server Management**

If a single business unit is to manage Policy Servers centrally, consider that central Policy Server management:

- Can facilitate a consistent implementation of CA Single Sign-On core objects, policies, and application objects because all CA Single Sign-On administrators are located in the same business unit.
- Can make the management of these objects easier because all CA Single Sign-On administrators are located in the same business unit.

  Note: As illustrated, individual business units can continue to manage the CA Single Sign-On Agents protecting their applications.

- Can simplify the CA Single Sign-On infrastructure. Central management can result in fewer Policy Server and policy stores to manage and upgrade.
- Lets administrators monitor CA Single Sign-On performance centrally.

Determine how to Manage CA Single Sign-on WSS Agents

If you have several CA Single Sign-on WSS Agents which will be configured identically, then using an Agent Configuration object on the Policy Server will make managing those Agents easier. A single Agent configuration object can be shared among an unlimited number of CA Single Sign-on WSS Agents. Configuration changes made on the Policy Server are automatically applied to any CA Single Sign-on WSS Agents which use the configuration object.

Capacity Planning

This content describes how to plan sufficient capacity in your CA Single Sign-Onimplementation to achieve high availability and performance.

Capacity Planning Introduced

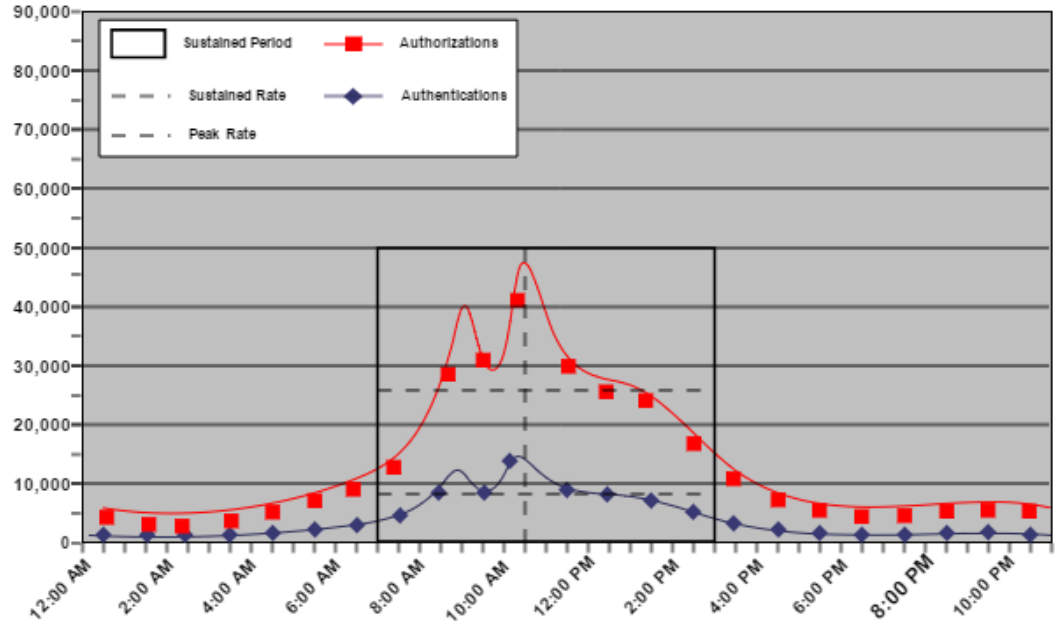
Planning a deployment with performance in mind is the first step to maintaining high enterprise availability and performance standards. A good approach is to estimate the number of expected authentications and authorizations CA Single Sign-On must handle per application. The following general factors influence performance:

- Sustained authentication and authorization rates. The rate at which users authenticate to an application and request protected resources fluctuates throughout your business day. Some periods can generate relatively few authentication requests, and therefore relatively few authorization requests, while others generate more. The sustained authentication and authorization rates represent a sustained period during which CA Single Sign-On must service an average number of authentication and authorization requests.

- Peak authentication and authorization rates. During sustained periods of activity, user activity may spike. The peak authentication and authorization rates represent a period during which CA Single Sign-On must service the highest number of authentication and authorization requests.

> **Note:** Although a number of other factors can influence performance, such as performance tuning and network bandwidth, the previous factors can help you make informed decisions when implementing Policy Servers and Agents, and when determining if existing user stores can handle the anticipated workload.

The following graphic illustrates how authentication and authorization rates fluctuate throughout the day, are sustained for a specific period, and peak within that period:



Note: Authenticating and authorizing users results in a number of reads, and if Password Policies are enabled, writes, to a user store. Determining sustained and peak rates helps you determine the load under which your user stores must operate to service Policy Server requests.

Estimate a Sustained Authentication Rate

Estimating the sustained authentication rate of an application is the process of determining:

- How the total number of authentication requests fluctuate throughout your business day
- How the authentication requests translate into requests per second.

Estimate Daily Authentications

What is the estimated number of daily authentications for the application?

The number of users directly affect daily authentications (authentication load). When users log into the application, CA Single Sign-On authenticates them. Therefore, think of the authentication load of the application as the total logins per day.

Note: When determining the authentication load, we recommend beginning with an evaluation interval of 24 hours. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

All users logging into the application each day is unlikely, so estimating total logins begins with determining the percentage of users that log in once a day, which the following represents:

(*total_users* * *percentage_users*) * (*number_of_logins*) = *daily_logins*

- **total_users**
  Represents the total number of users with access to the application.
- **percentage_users**
  Represents the percentage of users who log in the same number of times per day.
- **number_of_logins**
  Represents the number of times the particular set of users login.
- **daily_logins**
  Represents the number of logins the particular set of users creates.

Example 1: The company has 100,000 users, 75 percent of which log in once a day.

(100,000 * 0.75) x (1) = 75,000 logins

However, some users logging into the application two or more times a day is more likely.

Example 2: The company has 100,000 users, 5 percent of which log in twice a day and 1 percent of which log in three times a day.

(100,000 * 0.05) x (2) = 10,000 logins

(100,000 * 0.01) x (3) = 3,000 logins

The total logins per day are the sum of each of the login calculations.

Example 3: The company has 100,000 users:

- 75 percent of which log in once a day, creating 75,000 logins.
- Five percent of which log in twice a day, creating 10,000 logins.
- One percent of which log in three times a day, creating 3,000 logins.

The authentication load for the portal application is 88,000 logins.

Note: The percentage of users logging in does not have to equal 100 percent because all users will not log into the application each day.

The following table illustrates each of the previous examples:

Total Users	Percent of Total Users	Logins Per Day	Logins
100,000	75	1	75,000

Total Users	Percent of Total Users	Logins Per Day	Logins
100,000	5	2	10,000
100,000	1	3	3,000
Authentication Load			**88,0000**

The company uses the authentication load to estimate the sustained authentication rate.

Estimate a Sustained Authentication Rate

What is the sustained authentication rate for the application?

The sustained authentication rate is based on the authentication load. Specifically, when and at what rate the authentications occur. The chance that the authentication load is uniformly spread across your business day is unlikely. Rather, the rate at which requests occur fluctuates, remaining between the lowest and highest (peak) levels for a sustained period. Estimating the sustained authentication rate is the process of identifying a sustained period during which the system is servicing an average amount of authentication requests.

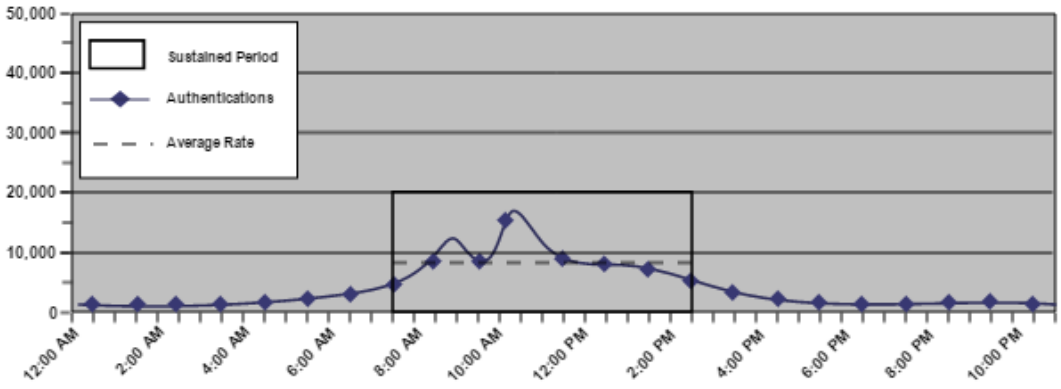
When estimating a sustained authentication rate, we recommend using the daily authentication load to determine:

- The rate at which the authentication requests occur throughout your business day.

  Note: We recommend beginning with an evaluation period of 24 hours, broken down into one-hour increments. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

- The sustained period during which the system is servicing an average number of authentication requests.
- The approximate number of authentication requests that occur during the sustained period.

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of authentication requests, per second, that CA Single Sign-On must service to maintain the average rate at which users authenticate, which the following represents:

(*authentication_load* * *percentage_of_authentication_requests*) / *number_of_sustained_hours* / 3600 = *sustained_authentication_rate*

- **authentication_load**
  Represents the number of daily authentications for the application.

- **percentage_of_authentication_requests**
  Represents the percentage of authentication requests that occur when the system is operating at sustained levels.
  Example: If the authentication load is 50,000 logins, and 32,000 logins occur during the sustained period, then the value is 64percent (0.64)

- **number_of_sustained_hours**
  Represents the number of hours in which the system is operating at the sustained level.

  
  Note: 3,600 represents the number of seconds in an hour.

- **sustained_authentication_rate**
  Represents the number of authentication requests, per second, that CA Single Sign-On must service during the period of sustained activity.

Example: Estimate the Sustained Authentication Rate

The company has determined that their application portal has an authentication load of 88,000 logins. The application portal is available to customers 24 hours a day, seven days a week. Using system activity reports to break down a typical day results in the following metrics:

- The system is operating at sustained levels for approximately five hours (9:00 AM - 2:00 PM).
- During sustained levels, approximately 9,000 authentications requests occur per hour.
- Approximately 45,000 (9,000 * 5) authentication requests, or 51 percent (45,000 / 88,000) of the daily authentication load, occur during these hours.

(88,000 * 0.51) / 5 / 3600 = 2.49 authentications per second.

The portal application has a sustained authentication rate of 2.49 authentications per second.

Estimate a Peak Authentication Rate

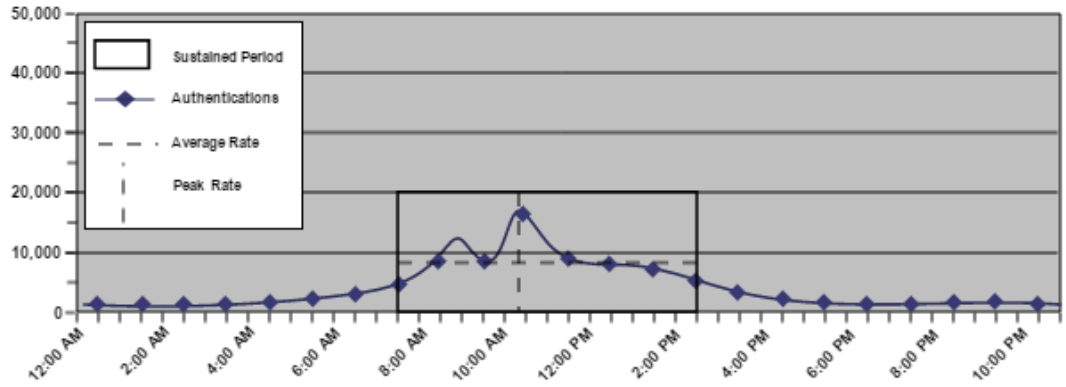
What is the peak authentication rate for the application?

The peak authentication rate is based on the sustained authentication rate, specifically, when and at what rate the system is operating at peak levels. Estimating the peak authentication rate is the process of identifying when the system is servicing the highest level of authentication requests.

When estimating the peak authentication rate, we recommend using the metrics you gathered when determining the sustained authentication rate to determine:

- The hour when the system is servicing the highest number of authentication requests
- The approximate number of authentication requests that occur during this period

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of authentication requests, per second, that CA Single Sign-On must service to maintain the peak rate at which users authenticate, which the following represents:

(*authentication_load* x *percentage_of_transactions*) / *number_of_hours* / 3600 = *peak_authentication_rate*

Note: This rate is based on the single busiest hour. There can be periods when the peak authentication rate exceeds the hourly calculation.

- **authentication_load**
  Represents the number of daily authentications for the application.
- **percentage_of_transactions**
  Represents the percentage of transactions that occur when the system is operating at peak levels.
- **number_of_hours**
  Represents the number of hours in which the system operates at peak levels.

  Note: 3,600 represents the number of seconds in an hour.
- **peak_authentication_rate**
  Represents the peak authentication rate for the application.

Example: Estimate the Peak Authentication Rate

The company has determined that their portal application has a daily authentication load of 88,000 logins. System activity reports detail that during the single busiest hour of the day 18,000 authentication requests occur. This number represents approximately 20 percent of the authentication load:

18,000 / 1 / 3600 = 5 authentications per second

The portal application has a peak authentication rate of five authentications per second.

Note: This example is based on the single busiest hour. There can be periods when the peak authentication rate during the hour exceeds five authentications per second.

Estimate a Sustained Authorization Rate

Estimating the sustained authorization rate for the application is the process of determining:

- How the total number of authorization requests fluctuate throughout your business day.
- How the authorization requests translate in to requests per second.

Estimate Daily Authorizations

What is the estimated number of daily authorizations for the application?

The number of total logins (authentication load) and the number of page "hits" each authenticated user makes directly affects the number of daily authorizations (authorization load). A web page "hit" usually requires an authorization. Therefore, think of the authorization load of an application as total authorizations per day.

Note: When estimating the authorization load, we recommend that you begin with an evaluation interval of 24 hours. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding usage throughout the year.

All users requesting the same number of pages per login is unlikely, so calculating total authorizations begins with determining the percentage of logins that generate one page hit, which the following represents:

*authentication_load * percentage_of_authenticated_users * page_visits = daily_authorizations*

- **authentication_load**
  Represents the estimated number of daily authentications for the application.
- **percent_of_authenticated_users**
  Represents the percentage of authenticated users that visit the same number of pages after login.

- **page_visits**
  Represents the number of pages a particular set of authenticated users visits after login.

  Note: A page can result in multiple GET/POST because it contains multiple objects. The total number of authorizations per page is the number of GET requests, plus the number of POST requests, minus the number of extensions the Web Agent ignores. Each of the following examples assume that a page visit generates one GET/POST.

- **daily_authorizations**
  Represents the number of authorizations a particular set of authenticated users require.

Example 1: Estimate Daily Authorizations

As detailed in Estimate Daily Authentications, the portal application has an authentication load of 88,000 logins. Twenty-five percent of which visit one page after login:

88,000 * 0.25 * 1 = 22,000 authorizations

However, some logins generating more than one page hit is more likely.

Example 2: Estimate Daily Authorizations

The portal application has an authentication load of 88,000 logins:

- 50 percent of which visit 10 pages after login.
- 25 percent of which visit 15 pages after login.

88,000 * 0.5 * 10 = 440,000 authorizations

88,000 * 0.25 * 15 = 330,000 authorizations

The total authorizations per day (authorization load) is the sum of each of the authorization calculations.

Example 3: Estimate Daily Authorizations

The portal application has an authentication load of 88,000 logins:

- 25 percent of which generate one page hit after login, creating 22,000 authorizations.
- 50 percent of which generate 10 page hits after login, creating 440,000 authorizations.
- 25 percent of which generate 15 page hits after login, creating 330,000 authorizations.

Note: The percentage of authenticated users must equal 100 percent because each authenticated user generates at least one page hit.

Therefore, the authorization load for the portal application is 792,000.

The following table illustrates each of the previous examples:

Page Hits	Percent of Total Logins	Authentication Load	Authorizations
1	25	88,000	22,000
10	50	88,000	440,000
15	25	88,000	330,000
Authorization Load			**792,000**

The company uses the authorization load to estimate the sustained authorization rate.

Estimate a Sustained Authorization Rate

What is the sustained authorization rate for the application?

The sustained authorization rate is based on the authorization load, specifically, when and at what rate the authorizations occur. The chance that the authorization load is uniformly spread across your business day is unlikely. Rather, the rate at which requests occur fluctuates, remaining between the lowest and highest (peak) levels for a sustained period. Estimating the sustained authorization rate is the process of identifying a sustained period during which the system is servicing an average amount of authorization requests.

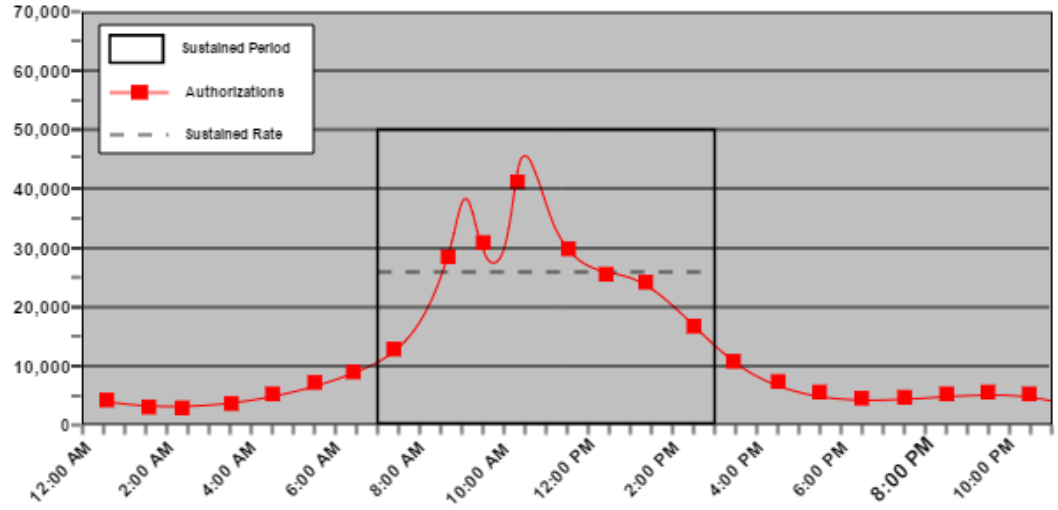
When estimating a sustained authorization rate, we recommend that you use the daily authorization load to determine:

- The rate at which the authorization requests occur throughout your business day.

  Note: We recommend beginning with an evaluation period of 24 hours, broken down into one-hour increments. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

- The sustained period during which the system is servicing an average number of authorization requests.
- The approximate number of authorization requests that occur during the sustained period.

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of authorization requests, per second, that CA Single Sign-On must service to maintain the average rate at which authorization requests occur, which the following represents:

(*authorization_load* * *percentage_of_authorization_requests*) / *number_sustained_hours* / 3600 = *sustained_authorization_rate*

- **authorization_load**
  Represents the number of daily authorizations for the application.

- **percentage_of_authorization_requests**
  Represents the percentage of authorization requests that occur when the system is operating at sustained levels.
  Example: If the authorization load is 500,000 requests, and 320,000 requests occur during the sustained period, then the value is 64 percent (0.64)

- **number_of_sustained_hours**
  Represents the number of hours in which the system is operating at the sustained level.

  Note: 3,600 represents the number of seconds in an hour.

- **sustained_authentication_rate**
  Represents the number of authorization requests, per second, that CA Single Sign-On must service during the period of sustained activity.

Example: Estimate a Sustained Authorization Rate

As detailed in Estimate Daily Authorizations (see page 64), the portal application has an authorization load of 792,000. The application portal is available to customers 24 hours a day, seven days a week. Using system activity reports to break down a typical day results in the following metrics:

- The system is operating at sustained levels for approximately five hours (9:00 AM - 2:00 PM)
- During sustained levels, approximately 75,000 authorization requests occur per hour.
- Approximately 375,000 (75,000 * 5) authorization requests, or 47 percent (375,000 / 792,000) of the daily authorization load, occur during these hours.

(762,000 * 0.47) / 5 / 3600 = 19.90 authorizations per second

The portal application has a sustained authorization rate of 19.90 authorizations per second.

Estimate a Peak Authorization Rate

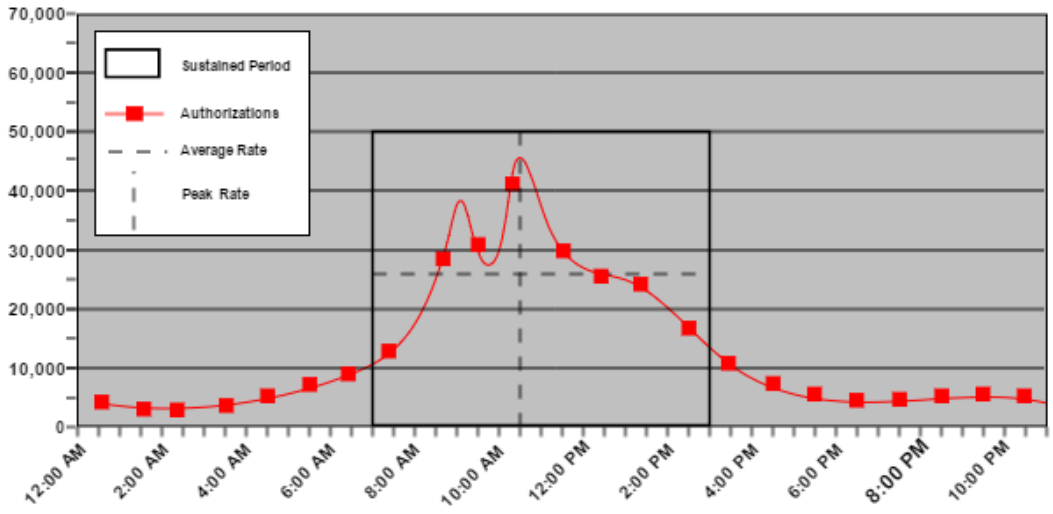
What is the peak authentication rate for the application?

The peak authorization rate is based on the sustained authorization rate, specifically, when and at what rate the system is operating at peak levels. Estimating the peak authorization rate is the process of identifying when the system is servicing the highest level of authorization requests.

When estimating the peak authorization rate, we recommend using the metrics that you gathered when determining the sustained authorization rate to determine:

- The hour the system is servicing the highest number of authorization requests
- The approximate number of authorization requests that occur during this period

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of authentication requests, per second, that CA Single Sign-On must service to maintain the peak rate at which users authenticate, which the following represents:

(*authorization_load* * *percentage_of_transactions*) / *number_of_hours* / 3600 = *peak_authorization_rate*

> **Note**: This rate is based on the single busiest hour. There can be times when the peak authorization rate exceeds the hourly calculation.

- **authorization_load**
  Represents the number of daily authorizations for the application.
- **percentage_of_transactions**
  Represents the percentage of transactions that occur when the system is operating at peak levels.
- **number_of_hours**
  Represents the number of hours in which the system is operating at peak levels.
- **peak_authorization_rate**
  Represents the peak authorization rate for the application.

Example: Estimate a Peak Authorization Rate

As detailed in Estimate Daily Authorizations, the portal application has an authorization load of 792,000. System activity reports detail that during the single busiest hour of the day, 260,000 authorization requests occur. This number represents approximately 33 percent of the authorization load.

(792,000 * 0.33) / 1 / 3600 = 72.6 authorizations per second

The portal application has a peak authentication rate of 72.6 authorizations per second.

Web Services Security Capacity Planning

This content describes how to plan sufficient capacity in your Web Services Security implementation to achieve high availability and performance.

Capacity Planning Introduced

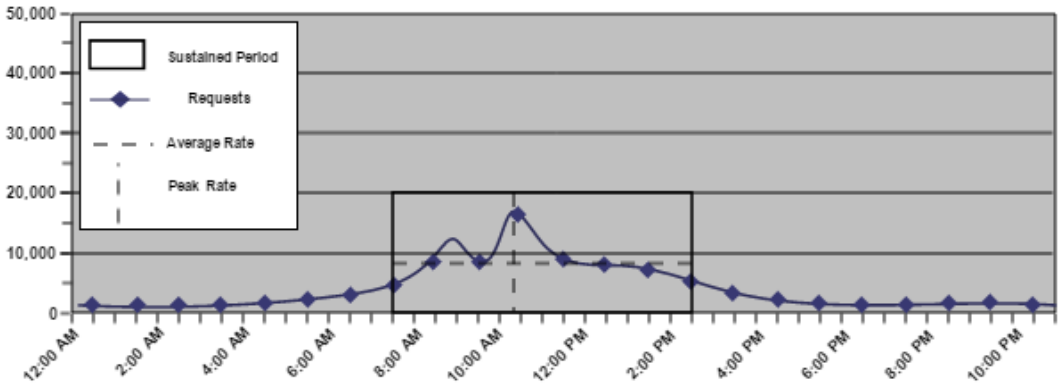
Planning a CA Single Sign-On Web Services Security deployment with performance in mind is the first step to maintaining high enterprise availability and performance standards. A good approach is to estimate the number of expected requests CA Single Sign-On must handle per web service. The following are the most significant factors that influence CA Single Sign-On performance:

- Sustained request rates. The rate at which web service clients send requests to protected web service resources fluctuates throughout your business day. Some periods can generate relatively few requests, and therefore require relatively few authentications and authorizations, while others generate more. The sustained request rates represent a sustained period during which CA Single Sign-On must service an average number of authentication and authorization requests.

  Note: Each web service request triggers one authentication and one authorization event.

- Peak request rates. During sustained periods of activity, web service client activity may spike. The peak request rates represent a period during which CA Single Sign-On must service the highest number of authentication and authorization requests.

The following graphic illustrates how request rates fluctuate throughout the day, are sustained for a specific period, and peak within that period:



Note: Authenticating and authorizing requests results in a number of reads from a user store. Determining sustained and peak rates helps you determine the load under which your user stores must operate to service Policy Server requests.

Capacity Planning Use Case

The purpose of the following use case is to illustrate how example.com (http://example.com), a fulfillment service organization approaches capacity planning by modeling the usage of their order fulfillment web service. The use case is referenced throughout this chapter for examples.

The company is planning to deploy CA Single Sign-On Web Services Security to protect its web service. The company has 10,000 users in a single user store.

Some web service clients send a single status request to the inventory operation of the web service once a day while others may send as many as four batched order requests per day to the fulfillment operation.

How to Estimate a Sustained Request Rate

Estimating the sustained request rate for a web service is the process of determining:

- How the total number of requests fluctuate throughout your business day
- How the requests translate into requests per second.

Estimate Daily Requests

What is the estimated number of daily requests for the web service?

The number of web service clients directly affect daily requests (request load). When web service clients send a request to the web service, CA Single Sign-On authenticates them. Therefore, think of the request load of the web service as the total requests per day.

Note: When determining the request load, we recommend beginning with an evaluation interval of 24 hours. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

All web service clients sending requests to the web service each day is unlikely, so estimating total requests begins with determining the percentage of web service clients that send a request once a day, which the following represents:

(*total_clients* * *percentage_clients*) * (*number_of_requests*) = *daily_logins*

- **total_clients**
  Represents the total number of clients with access to the application.
- **percentage_clients**
  Represents the percentage of clients that send requests the same number of times per day.
- **number_of_requests**
  Represents the number of times the particular set of clients send requests.

- **daily_logins**
  Represents the number of logins the particular set of clients creates.

Example

The company has 10,000 users, 60 percent of which send an inventory status request once a day.

(10,000 * 0.6) x (1) = 6,000 logins

Additionally, 30 percent of users send one order fulfillment request per day, 20 percent of users send two order fulfillment requests a day, 10 percent send three order fulfillment requests a day, and 10 percent send four order fulfillment requests a day.

(10,000 * 0.3) x (1) = 3,000 logins

(10,000 * 0.2) x (2) = 4,000 logins

(10,000 * 0.1) x (3) = 3,000 logins

(10,000 * 0.1) x (4) = 4,000 logins

The total requests per day are the sum of each of the request calculations. The request load for the fulfillment web service is therefore 20,000 logins.

Note: The percentage of clients making requests is not necessarily equal to 100 percent because not all clients will necessarily send a request to the service each day.

The company uses the request load to estimate the sustained request rate.

Estimate a Sustained Request Rate

What is the sustained request rate for the web service?

The sustained request rate is based on the request load. Specifically, when and at what rate the requests occur. The chance that the request load is uniformly spread across your business day is unlikely. Rather, the rate at which requests occur fluctuates, remaining between the lowest and highest (peak) levels for a sustained period. Estimating the sustained request rate is the process of identifying a sustained period during which the system is servicing an average number of requests.

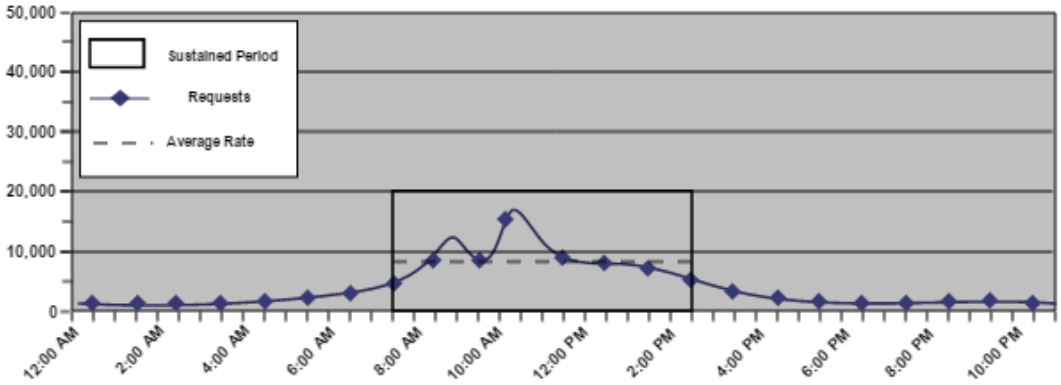
When estimating a sustained request rate, we recommend using the daily request load to determine:

- The rate at which the requests occur throughout your business day.

Note: We recommend beginning with an evaluation period of 24 hours, broken down into one-hour increments. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

- The sustained period during which the system is servicing an average number of requests.
- The approximate number of requests that occur during the sustained period.

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of requests, per second, that CA Single Sign-On must service to maintain the average rate at which users authenticate, which the following represents:

(*request_load* * *percentage_of_requests*) / *number_of_sustained_hours* / 3600 = *sustained_request_rate*

- **request_load**
  Represents the number of daily requests for the application.
- **percentage_of_requests**
  Represents the percentage of requests that occur when the system is operating at sustained levels.
  Example: If the request load is 5,000 logins, and 3,000 logins occur during the sustained period, then the value is 64percent (0.64)
- **number_of_sustained_hours**
  Represents the number of hours in which the system is operating at the sustained level.

  Note: 3,600 represents the number of seconds in an hour.
- **sustained_request_rate**
  Represents the number of requests, per second, that CA Single Sign-On must service during the period of sustained activity.

Example: Estimate the Sustained Request Rate

The company has determined that their web service has a request load of 2,000 logins. The web service is available to customers 24 hours a day, seven days a week. Using system activity reports to break down a typical day results in the following metrics:

- The system is operating at sustained levels for approximately five hours (9:00 AM - 2:00 PM).
- During sustained levels, approximately 2,500 requests occur per hour.
- Approximately 1,250 (250 * 5) requests, or 62.5 percent (1,250 / 2,000) of the daily request load, occur during these hours.

(2,000 * 0.625) / 5 / 3600 = 0.0694 requests per second.

The fulfillment web service has a sustained request rate of 0.694 requests per second.

Estimate a Peak Request Rate

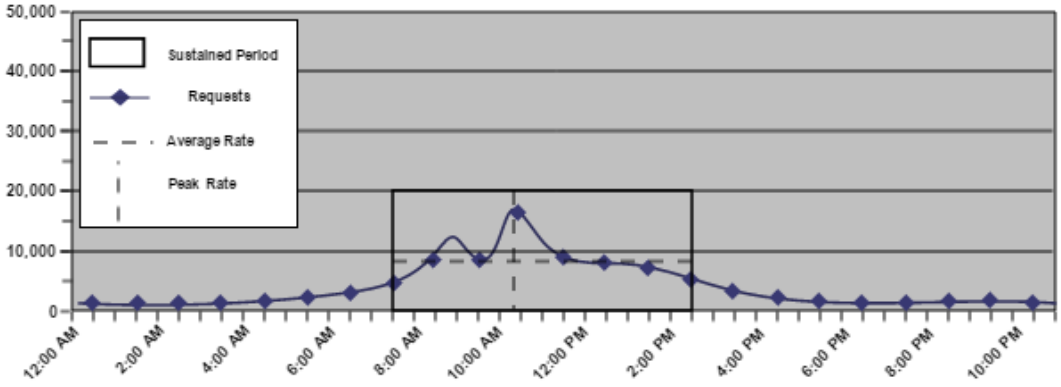
What is the peak request rate for the web service?

The peak request rate is based on the sustained request rate, specifically, when and at what rate the system is operating at peak levels. Estimating the peak request rate is the process of identifying when the system is servicing the highest level of requests.

When estimating the peak request rate, we recommend that you use the metrics you gathered when determining the sustained request rate to determine:

- The hour when the system is servicing the highest number of requests
- The approximate number of requests that occur during this period

The following figure is an example of these metrics:



Identifying these metrics helps you to estimate the number of requests, per second, that CA Single Sign-On must service to maintain the peak rate at which web service clients authenticate, which the following represents:

(*request_load* x *percentage_of_transactions*) / *number_of_hours* / 3600 = *peak_request_rate*

> **Note**: This rate is based on the single busiest hour. There can be periods when the peak request rate exceeds the hourly calculation.

- **request_load**
  Represents the number of daily requests for the web service.
- **percentage_of_transactions**
  Represents the percentage of transactions that occur when the system is operating at peak levels.
- **number_of_hours**
  Represents the number of hours in which the system operates at peak levels.

  > **Note:** 3,600 represents the number of seconds in an hour.
- **peak_request_rate**
  Represents the peak request rate for the application.

Example: Estimate the Peak Request Rate

The company has determined that their web service has a daily request load of 8,800. System activity reports detail that during the single busiest hour of the day 1,800 requests occur. This number represents approximately 20 percent of the request load:

1,800 / 1 / 3600 = 0.5 requests per second

The fulfillment web service has a peak request rate of five requests per second.

Note: This example is based on the single busiest hour. There can be periods when the peak request rate during the hour exceeds five requests per second.

Other Factors to Consider When Capacity Planning

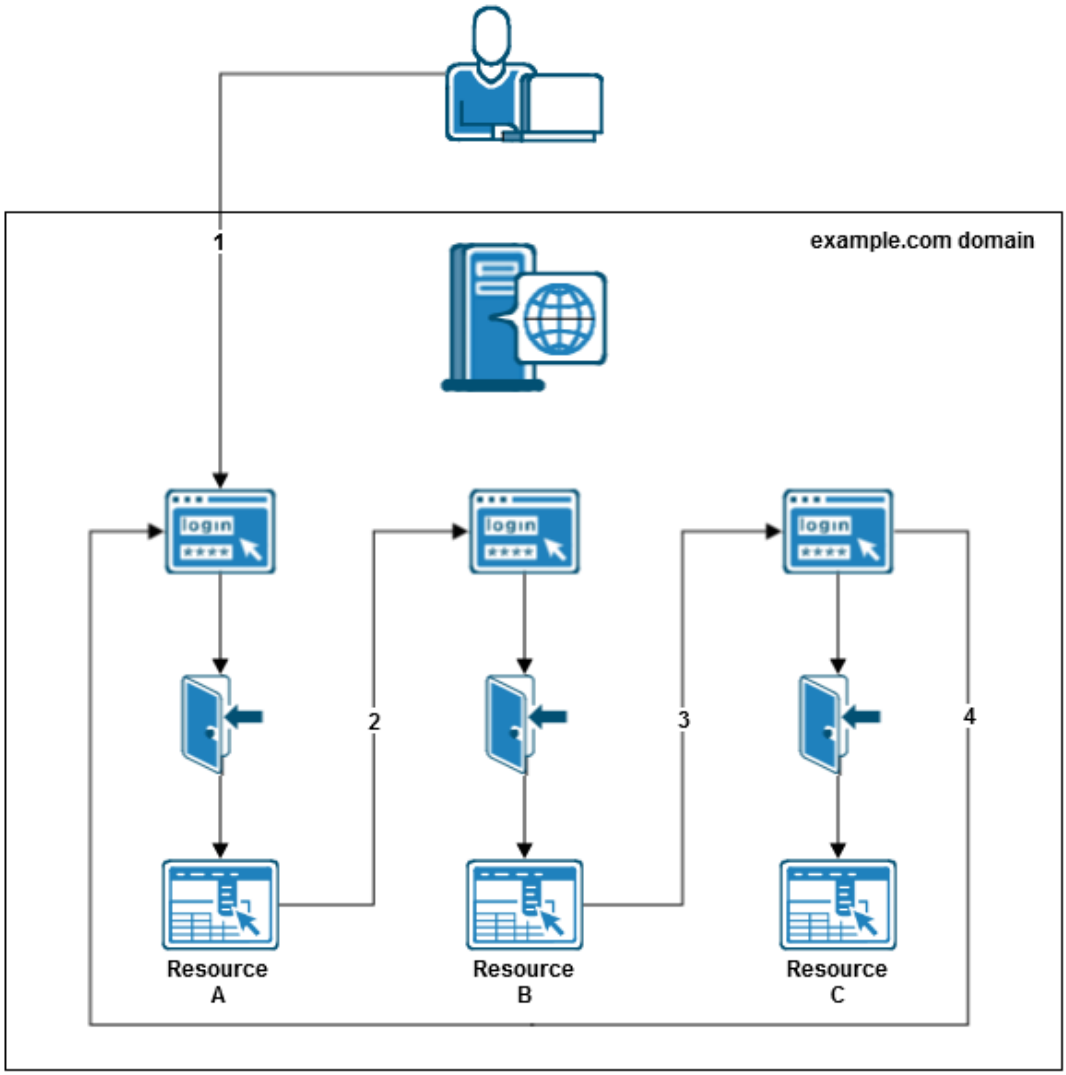
Although request rates are the most significant factors in CA Single Sign-On Web Services Security capacity planning decisions other factors can also influence CA Single Sign-On performance, in particular the authentication schemes used to protect your web services.

Also consider performance tuning and network bandwidth in your capacity planning process.

Security Zones

Security Zones are groups of resources in a single cookie domain that a CA Single Sign-On Web Agent protects. Users authenticate once, and can then access other resources in the zones (for which they are authorized) without being rechallenged.

Without Security Zones, users could possibly be challenged each time they access a protected resource in the same cookie domain; even if they have previously been authenticated by CA Single Sign-On for another resource in the cookie domain. The following illustration shows an example:

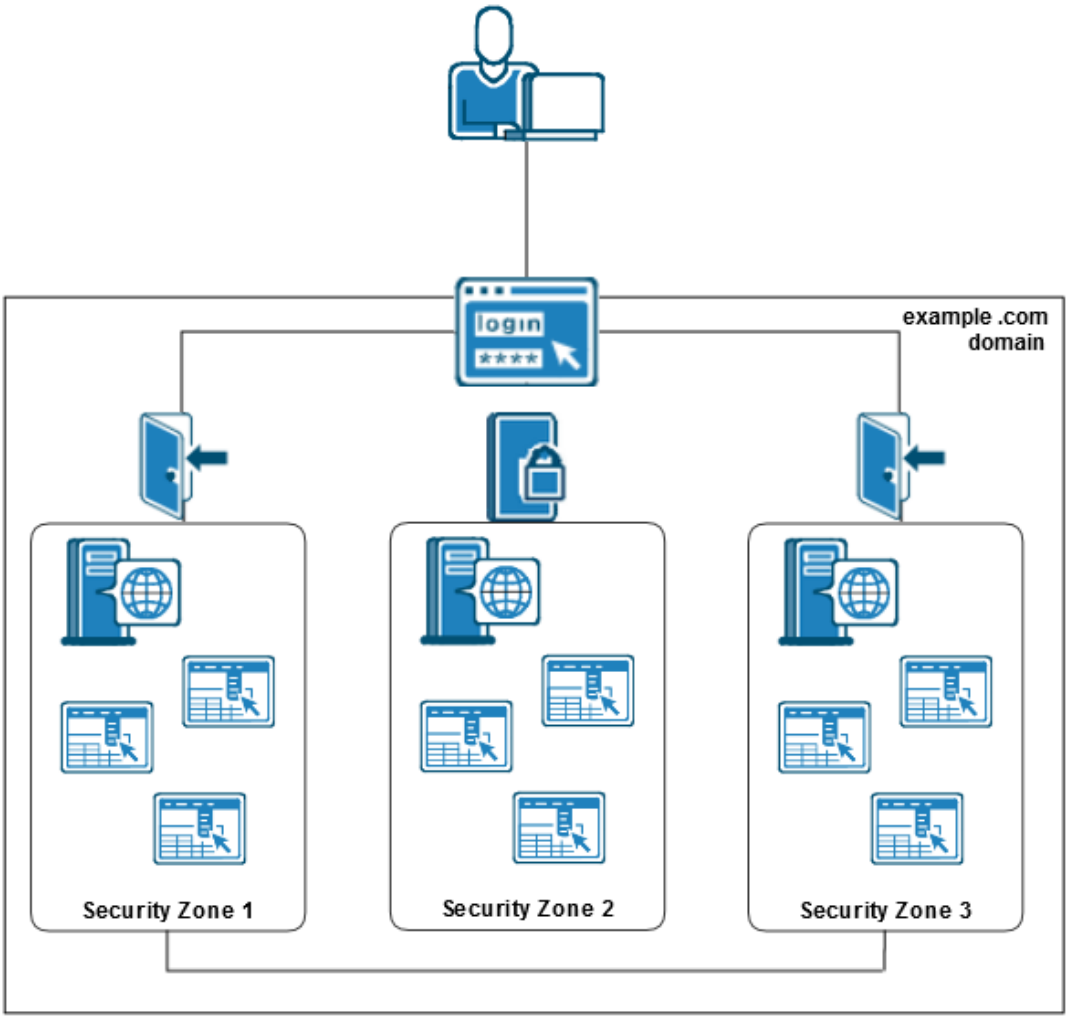


Consider implementing Security Zones in the following situations:

- You have several resources in a cookie domain, but you want to apply different access restrictions to those resources.

- You want to enable SSO between different resources in the same cookie domain.
- You want to create groups of resources that span several cookie domains and allow SSO between them.
- You have a large organization with a single cookie domain, and you use multiple instances of CA Single Sign-On to protect resources in your organization. Security Zones let you separate the resources to control access within the single cookie domain. Without Security Zones, the cookies used by one CA Single Sign-On instance could possibly overwrite the cookies of another CA Single Sign-On instance (cookie stomping) because the cookie domain name is the same for both instances.

The following illustration shows how Security Zones can be used so that only a single log in allows a user access to resources in Security Zones 1 and 3, but prevents access to unauthorized resources in Security Zone 2:'



Multiple Data Centers

Contents

CA Single Sign-On treats a global deployment the same as multiple data centers in the same continent. As such, factors outside of CA Single Sign-On affect the performance of a multi–data center deployment. The following key factors include:

- Network latency
- Resiliency

We recommend that you consider the following outside factors as you plan for a multi–data center deployment:

- Network infrastructure
- Application locations
- User locations
- User store vendors and their restrictions, such as the number of masters allowed

Best Practices

Consider the following when configuring data centers:

- Collocating the following components in each data center helps to reduce the effect network latency and resiliency has on CA Single Sign-On performance:
  - CA Single Sign-On Agents
  - Policy Servers
  - User stores

  Note: If a CA Single Sign-On feature, such as Password Services, requires a write–enabled store, we recommend having a write–enabled store in each data center.

- If all components cannot be in the same data center, we recommend at least collocating Policy Servers and user stores in the same data center.

Architectural Considerations

Consider the following architectural factors when planning for a CA Single Sign-On data center:

- CA Single Sign-On Password Services attempts to perform an LDAP write to the user account on every authentication.
- CA Single Sign-On follows LDAP write referrals when communicating with a read–only consumer directory.
- If you deploy a master policy store with replicated versions, consider using a local host file on the Policy Server host system (LDAP) or the ODBC data source to point Policy Servers to the local policy store. Using this method lets all Policy Servers share the same policy store and avoids the latency that can occur when all Policy Servers must communicate with the policy store over the wide area network (WAN).
- If you deploy master/consumer user stores, consider using a local host file on the Policy Server host system (LDAP) or the ODBC data source name (DSN) to point Policy Servers to the local consumer. Using this method lets all Policy Servers read the same user store and avoids the latency that can occur when all Policy Servers must read user account information over the WAN.

Example: Local Host Files Pointing Policy Servers to the Local Consumer User Store

Two geographically separated data centers include Policy Servers pointing to a consumer user store named myusers.

- The local consumer in data center one is available at 111.11.111.1
- The local consumer in data center two is available at 222.22.222.2

To point Policy Server to the local consumer

1. From the Policy Server host systems in data center one, use a local host file to map myusers to 111.11.111.1.
2. From the Policy Server host systems in data center two, use a local host file to map myusers to 222.22.222.2.

Multiple Data Center Use Cases

The purpose of the following use cases is to get you thinking about your CA Single Sign-On data centers in terms of network latency and resiliency. The use cases begin with a simple deployment and progress into more complex scenarios.

These use cases are intended to identify techniques that you can use as part of a global architecture and are not intended as a final architecture. Extrapolate the necessary infrastructure from these cases to configure data centers that best meet the needs of your organization.

All Components in One Data Center

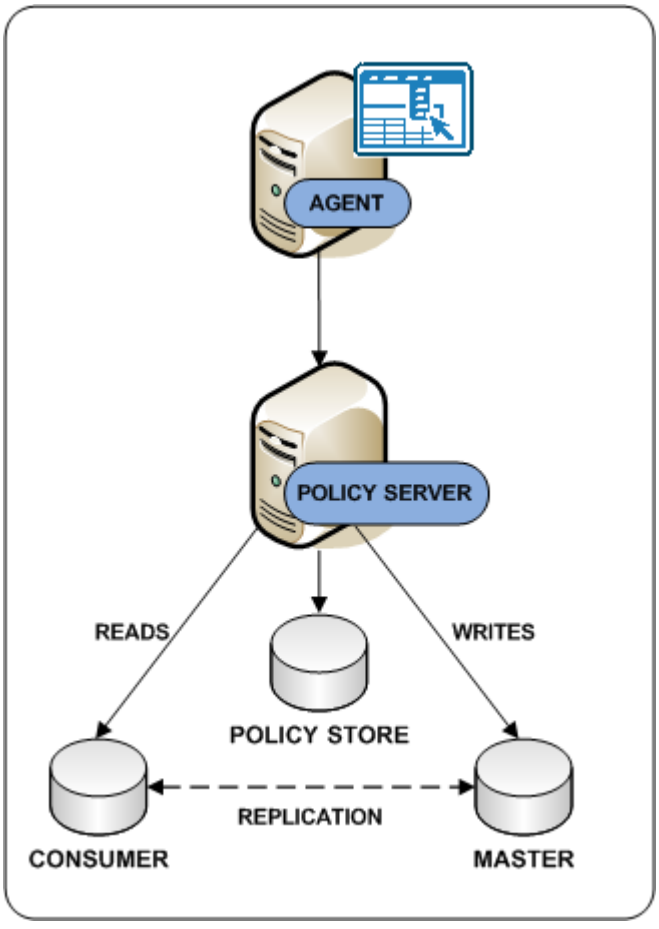
The simplest deployment includes all required CA Single Sign-On components in a single data center.

The following diagram illustrates:

- All applications in a single data center.
- A Policy Server writing to a master user store. CA Single Sign-On Password Services attempts to perform an LDAP write to the user account on every authentication.

> **Important!** For more information about multi–mastered LDAP user store support limitations, see the *Policy Server Release Notes*.

- CA Single Sign-On reading a consumer user store.



SM--Data Center

Consider the following:

- Although not illustrated, CA Single Sign-On supports database clusters that are configured for write and read–only transactions.
- You can configure multiple components in a data center for operational continuity, redundancy, and high availability.

All Components in Multiple Data Centers

You extend the CA Single Sign-On environment by deploying multiple data centers. The following factors can influence your decision to implement multiple data centers:

- The network infrastructure
- The location of applications
- The location of users

The following diagram illustrates:

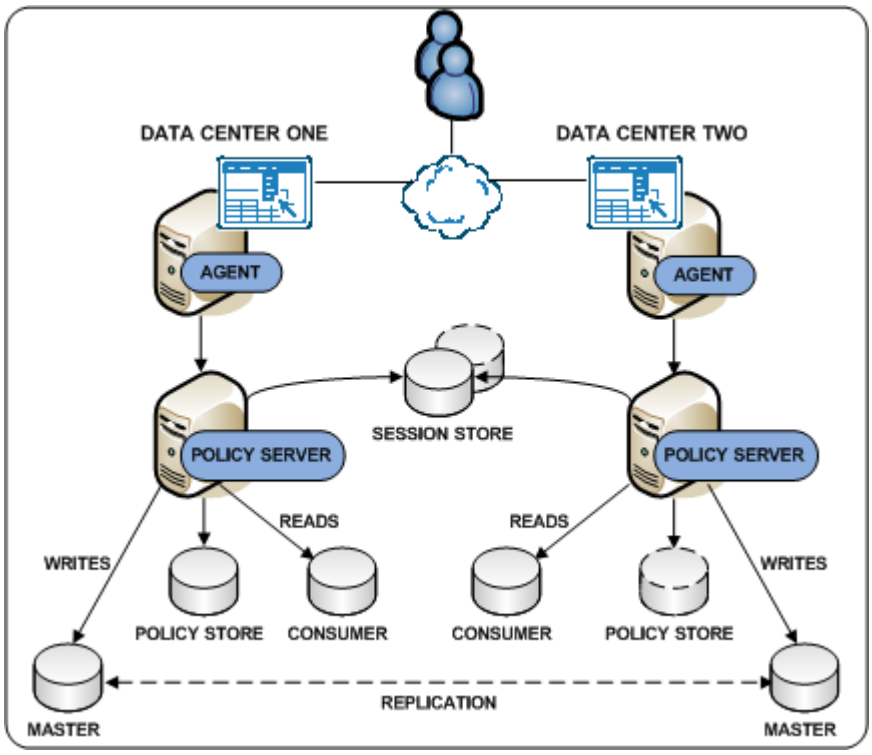
- Applications in multiple data centers
- Each data center using its own policy store. Data center one contains the primary policy store. Data center two contains the replicated version, as the dotted line details.

> **Note:** Every Policy Server in the deployment must share a common view into the same policy store. For more information about policy store redundancy, see Policy Server to Policy Store Communication (see page 27).

- Each data center using its own master/consumer user stores.

> **Important!** For more information about multi–mastered LDAP user store support limitations, see the *Policy Server Release Notes*.

- A centralized replicated session store to enable single sign–on between all applications.



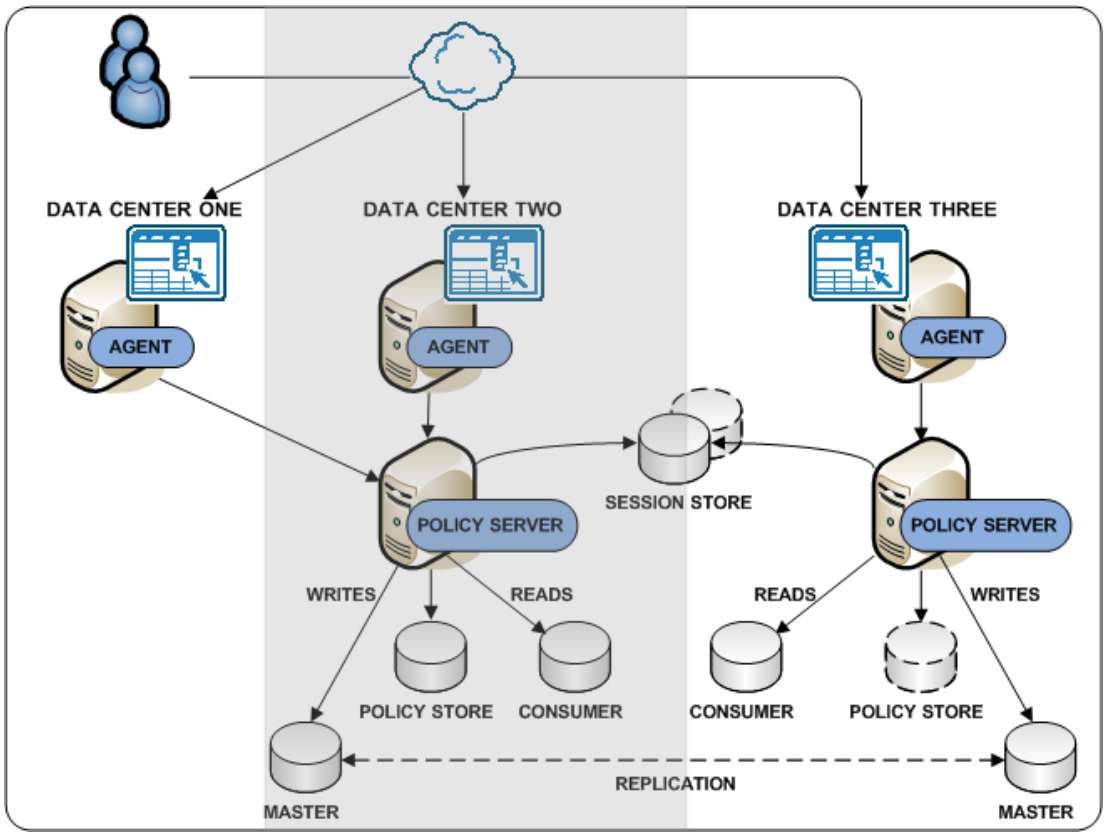
SM--All Components in Multiple Data Centers

CA Single Sign-On Agent Communicating Across a Data Center

If all components cannot be in the same data center, we recommend at least collocating Policy Servers and user stores in the same data center.

The following diagram illustrates:

- Applications in multiple data centers.
- Data center one only containing a web server with a CA Single Sign-On Agent. The agent communicates across the wide area network to a Policy Server in data center two.
- Data centers 2 and 3:
  - Sharing a common view into the policy store through a master/replicated policy store (see page 28).
  - Using their own master/consumer user stores (see page 80).
  - Using a centralized replicated session store to enable single sign–on between all applications.



SM--Web Agent Communicating Across Data Center

Policy Server Communicating Across a Data Center

If all components cannot be in the same data center, we recommend at least collocating Policy Servers and user stores in the same data center.

The following diagram illustrates:

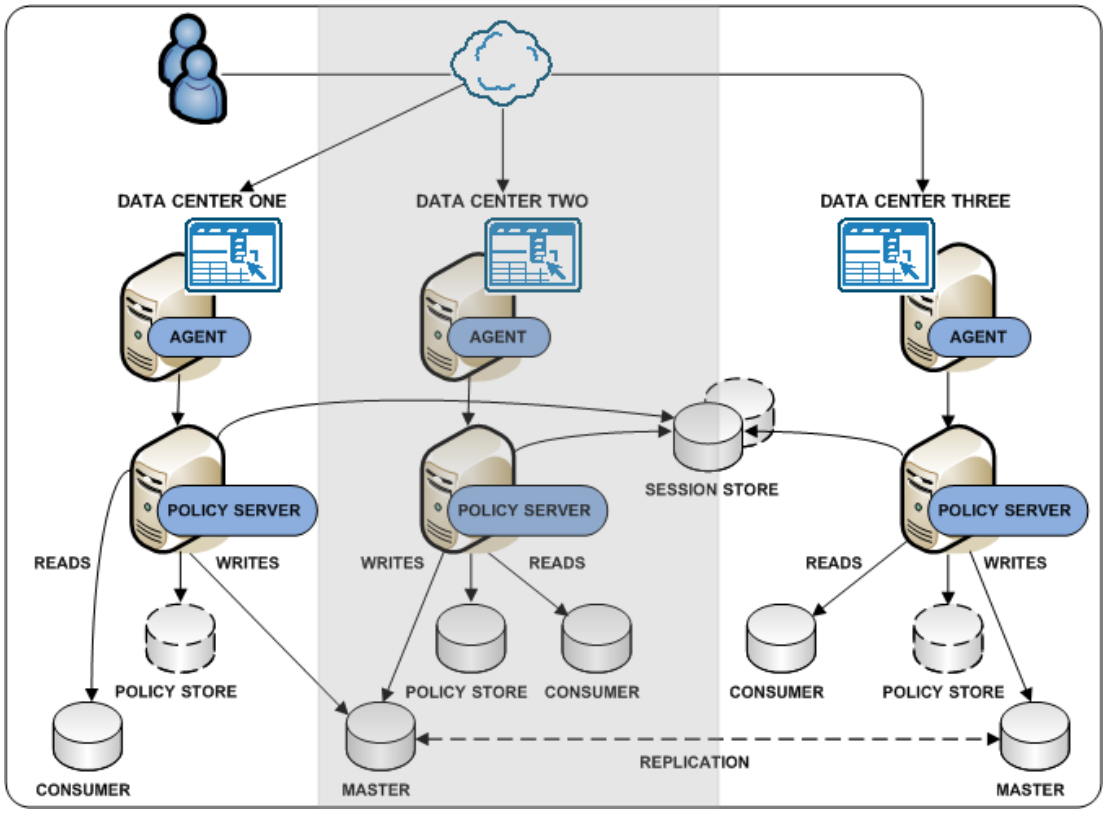
- Applications in multiple data centers.
- Data center 1 only containing an Agent and Policy Server. The Policy Server only communicates across the wide area network to perform LDAP writes to the master user store in data center 2.

  Important! We do not recommend configuring a Policy Server to communicate across the wide area network to perform LDAP reads and writes.

- All data centers:
  - Sharing a common view into the policy store through a master/replicated policy store (see page 28).
  - Using a centralized replicated session store to enable single sign–on between all applications.

- Data centers 2 and 3 using their own master/consumer user stores (see page 80).

> **Important!** For more information about multi–mastered LDAP user store support limitations, see the *Policy Server Release Notes*.



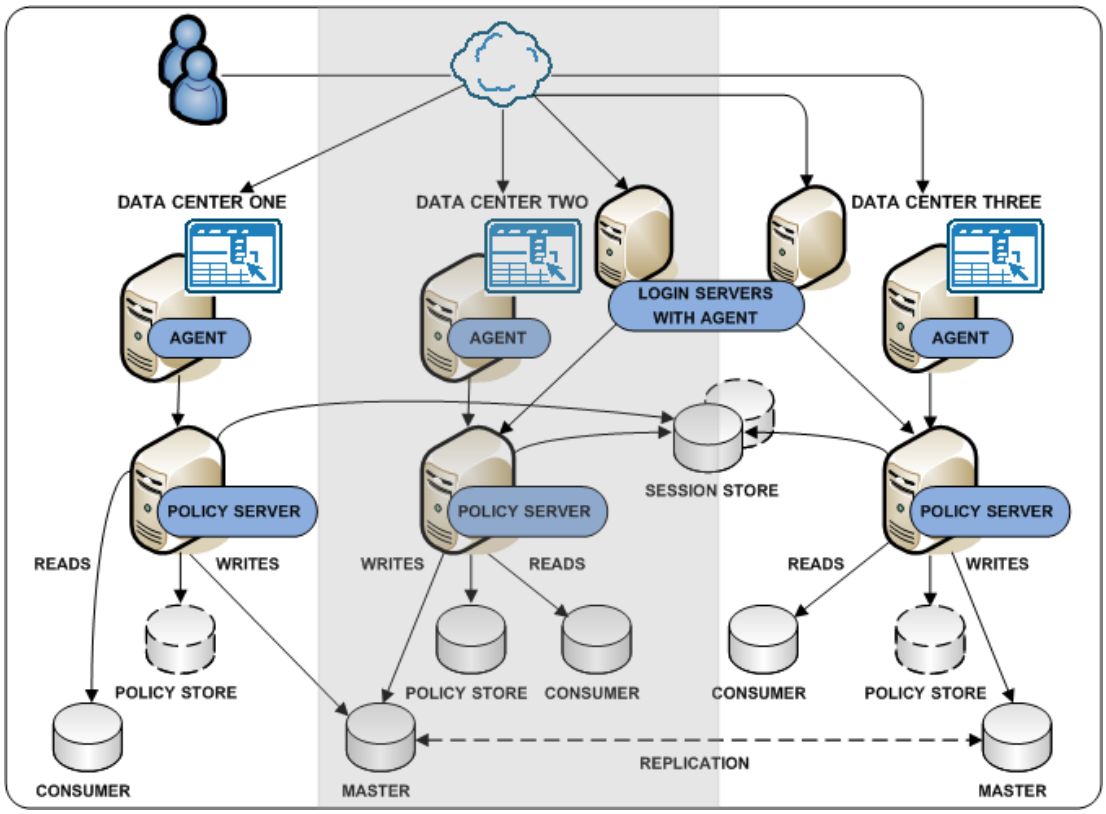
SM--Policy Server Communicating Across Data Center

Login Server Controlling User Store Writes

The location of LDAP writable masters can constrain a CA Single Sign-On deployment. Consider using one or more centralized login servers to eliminate requirements for writable masters in each data center.

The following diagram illustrates:

- A multiple data center deployment in which:
  - The Policy Server in data center one is communicating across the WAN to perform an LDAP write (see page 83).
  - The remaining data centers including all components (see page 81).
- A login server in data center two and data center three.



SM--Logon Servers in Data Centers

When users request access to a protected URL in data center one:

1. The Web Agent redirects the request to the logon server in data center two. The redirect is based on the authentication scheme that is protecting the resource.

2. The Policy Server in data center two authenticates the user and writes to the master user store.

3. The Policy Server creates a CA Single Sign-On session ticket and passes it back to the original protected URL.

4. A Web Agent places the CA Single Sign-On session ticket into a cookie. The Web Agent uses the cookie to handle subsequent authentication and authorization requests in the data center, until one of the following occurs:

   - The user requests another resource that requires additional credentials.

   - The session expires.

Authentication and a Centralized Login Server

Contents

A CA Single Sign-On deployment typically includes applications for which different authentication (login) requirements exist. These requirements can result in numerous login pages that the individual application owners must manage. Managing these login pages locally can introduce inconsistencies, such as page design and the presentation of error messages, that can affect the overall authentication experience.

We recommend managing login pages centrally to help:

- Create consistency across your applications. If a single CA Single Sign-On team owns all login pages, the team can implement them consistently and manage them easier.
- Minimize the number of login pages. Minimizing the number of entry points into applications creates the impression that users are logging into a centralized infrastructure, rather than individual applications.

Consider the following when configuring login pages:

- Identify applications that share the same authentication requirements and reuse the same login page.
- Use a centralized login server to host all login pages
- Configure login pages to inform users when:
  - They have failed to provide valid credentials.
  - Too many attempts have resulted in a failed authentication.

Centralize Login Pages

Application login requirements can range from basic user name/password authentication to forms–based authentication to digital certificates. If possible, we recommend:

- Managing all login pages from a central login server to avoid duplication on every web application.
- Managing all other system–wide resources, such as password services pages, error pages, and terms and conditions pages from a central server.

Managing login pages centrally is the process of identifying applications that share the same login requirements. Consider the following when configuring authentication:

- Try to avoid creating separate login pages for each application. As CA Single Sign-On adoption increases, managing a login page for every application may not be sustainable.

- Identify applications that share the same authentication requirements. If possible, use a single login page as an entry point into these applications.

Use a table similar to the following to group applications by authentication requirements:

Auth Scheme Name	Type	Login Page Server	Login Page URL

Example: Grouping applications by authentication requirements

A CA Single Sign-On environment protects ten applications:

- Five of the applications require forms-based authentication.
- Three of the applications require Windows-based authentication.
- Two of the applications require basic user name/password authentication.

Auth Scheme Name	Type	Login Page Server	Login Page URL
Auth1	Forms	login.acme.com	/login.asp
Auth2	Windows	login.acme.com	/smgetcrd.ntc
Auth3	Basic	login.acme.com	n/a

Best Practices

Consider the following when configuring login pages:

- Display an error message when a user fails to authenticate properly.
- Redirect users to a page that displays a message that the number of login attempts has been exceeded.
- We recommend using forms–based authentication to redirect users. If you are unable to use forms–based authentication, you can use the CA Single Sign-On OnAuthAttempt and OnAuthReject responses to redirect users.
  - If you configure forms–based authentication, consider creating a dynamic page, such as login.asp, to create a tighter integration with your existing infrastructure.
  - If creating a dynamic page is not possible, use the sample login FCC file (login.fcc) that is included as part of the Web Agent installation to configure a login FCC file. The default location for the sample file is *web_agent_home*\samples_default\forms. The forms directory is the default location for files that the Forms Credential Collector (FCC) processes.

- **web_agent_home**
  Specifies the Web Agent installation path.

- We recommend creating a separate directory on the Web Agent host system for all login pages. Using a location other than the forms directory helps to prevent the sample files from being accidentally overwritten.

- Display a custom logoff page after a user logs out successfully.

Login Page Use Cases

The purpose of the following use cases is to get you thinking about configuring CA Single Sign-On authentication.

These use cases reflect best practices and are intended to identify techniques that you can use as part of a global architecture. These use cases are not intended as a final architecture. Extrapolate the necessary infrastructure from these cases to configure login pages that best meet the needs of your organization.

Stand–Alone Login Page

In this use case, CA Single Sign-On directs users to a stand–alone login page when they request a protected resource. Specifically:

- A dynamic login page (login.asp) is deployed to the Web Agent host system.

- The dynamic login page is coded to:

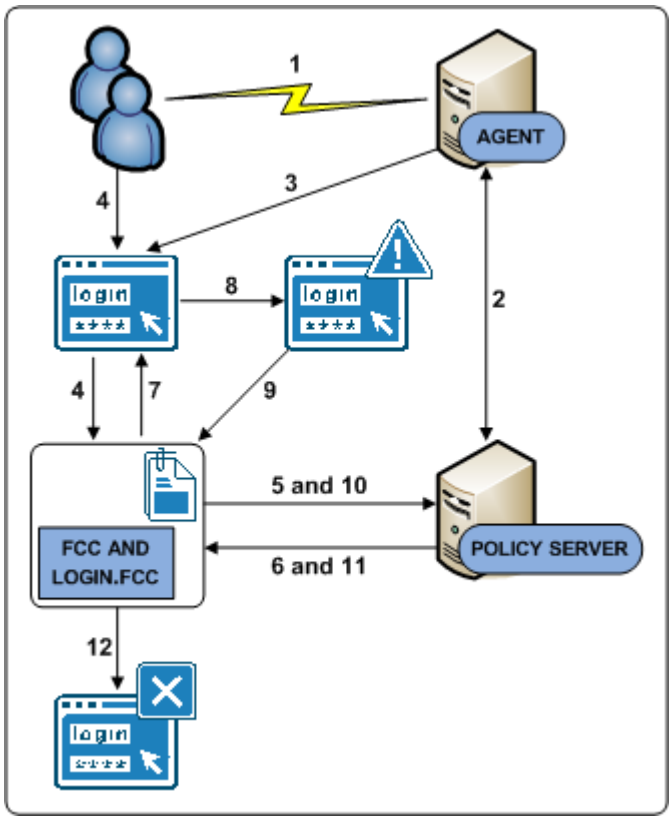
  - Post to a login FCC file (login.fcc).

  - Display an error message when the SMTRYNO cookie is present in the web browser of the user.

- The login FCC file is configured with an @directive (@smretries) to redirect users to a failed authentication page (login.unauth) after two failed authentication attempts.

- A CA Single Sign-On administrator has configured a form–based authentication scheme named Auth1. The target of Auth1 is login.asp.

The following diagram illustrates the authentication process for this use case:



SM--Stand Alone Dynamic Forms Login

1. A user requests a protected resource.
2. The Web Agent contacts the Policy Server, which determines that the resource is protected.
3. The Web Agent redirects the user request to login.asp.
4. The user submits invalid credentials. The credentials are posted to the login.fcc file and processed by the FCC.
5. The FCC forwards the credentials to the Policy Server.
6. The Policy Server determines that the credentials are invalid and notifies the FCC.
7. The FCC inserts the SMTRYNO cookie into the web browser of the user and redirects the user to the login page.
8. The login page refreshes with an error message. The error message states that invalid credentials were supplied and to try again.
9. The user submits invalid credentials. The credentials are posted to the login.fcc file and processed by the FCC.
10. The FCC forwards the credentials to the Policy Server.
11. The Policy Sever determines that the credentials continue to be invalid and notifies the FCC.

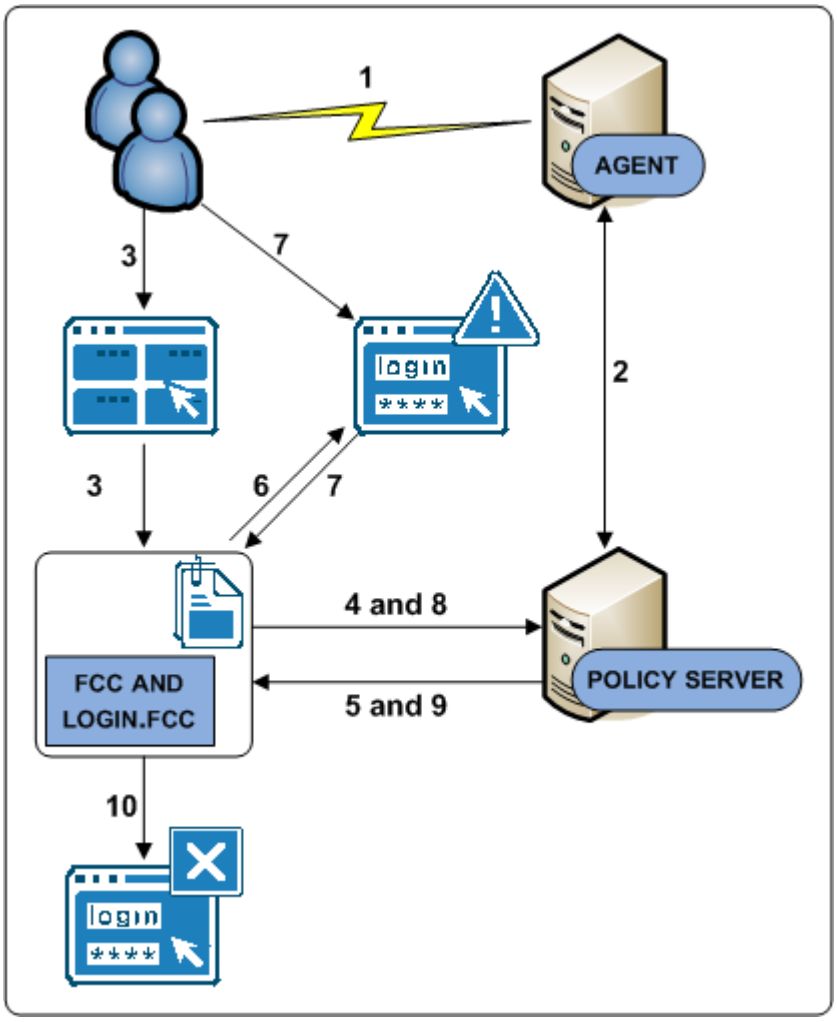
12. The user has exceeded the maximum number of failed authentication attempts and is redirected to a page that displays a failed authentication message.

Embedded Form on a Web Portal

In this use case, a form is embedded on a web portal home page. Users enter credentials in the form and are redirected to the protected resource upon authentication. Specifically:

- A web portal home page (portal.asp) includes an embedded form that prompts users for credentials. The home page:
  - Contains a target variable that points to the protected resource.
  - Posts to a login FCC file (login.fcc).
- A stand-alone login page (login.asp) is deployed to the Web Agent host system. If users try to access the protected resource directly, this page prompts users for credentials. The login page:
  - Posts to the login FCC file.
  - Displays an error message when the SMTRYNO cookie is present in the web browser of the user.
- The login FCC file is configured with an @directive (@smretries) to redirect users to a failed authentication page (login.unauth) after two failed authentication attempts.
- A CA Single Sign-On administrator has configured a form–based authentication scheme named Auth1. The target of Auth1 is login.asp.

The following diagram illustrates the authentication process for this use case:



SM--Embedded Forms Login

1. A user navigates to the web portal home page.
2. The Web Agent contacts the Policy Server, which determines that the resource is unprotected.
3. The user submits invalid credentials. The credentials are posted to the login.fcc file and processed by the FCC.
4. The FCC forwards the credentials to the Policy Server.
5. The Policy Server determines that the credentials are invalid and notifies the FCC.
6. The FCC inserts the SMTRYNO cookie into the web browser of the user and redirects the user to the login page. The login page appears with an error message. The error message states that invalid credentials were supplied and to try again.

> **Note:** Although not illustrated, if the user accessed the protected resource directly, the login page would appear without an error message because the web browser would not contain the SMTRYNO cookie.

7. The user submits invalid credentials. The credentials are posted to the login.fcc file and processed by the FCC.
8. The FCC forwards the credentials to the Policy Server.
9. The Policy Sever determines that the credentials continue to be invalid and notifies the FCC.
10. The user has exceeded the maximum number of failed authentication attempts and is redirected to a page that displays a failed authentication message.

Performance Tuning

This topics in this section (see Table of Contents to the left) describe how to tune your CA Single Sign-On environment to optimize performance.

Performance Tuning Introduced

The Policy Server evaluates and enforces access control policies by servicing three basic requests:

- IsProtected—is the requested resource protected?
- IsAuthenticated—did the user requesting the resource present credentials to establish an identity?
- IsAuthorized—is the authenticated user authorized to view the protected resource?

Servicing each of these requests creates transactions between CA Single Sign-On components. CA Single Sign-On performance tuning is the iterative process of increasing throughput and reducing latency by:

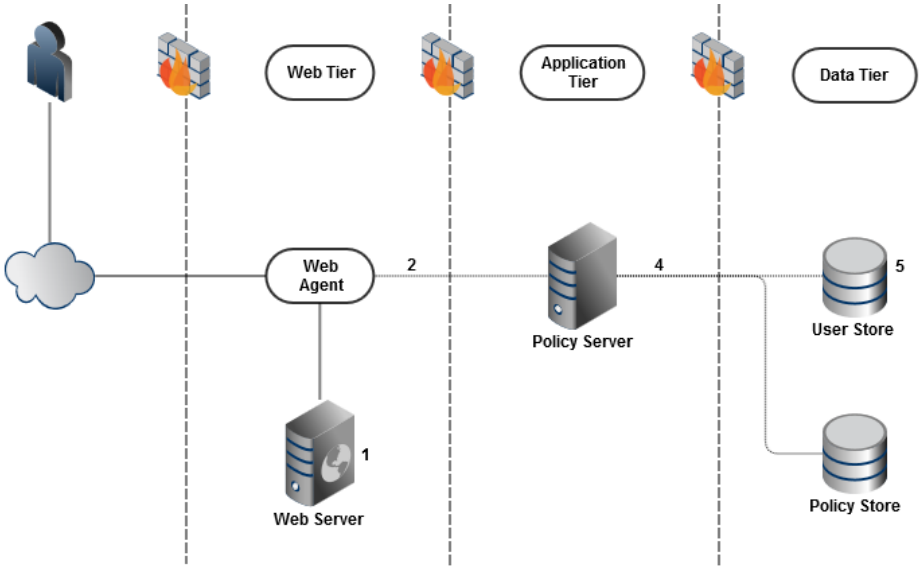
- Understanding where and when these transactions occur
- Identifying the CA Single Sign-On settings and features that affect performance
- Using third-party and CA Single Sign-On tools to measure performance and identify infrastructure bottlenecks

A good strategy is to examine performance factors in the web, application, and data tiers.

Note: CA Single Sign-On is middleware and is not deployed independently. The following sections focus on tuning CA Single Sign-On components in the Web and Application tiers, but not how to tune the actual Web, Application, or Data tiers themselves. See your vendor–specific documentation for more information about tuning the web servers, directory servers, and databases in your environment.

Performance Tuning Roadmap

Performance tuning is an iterative process, and as such, it is important to address the Web, Application, and Data tiers on an individual basis to understand how each can affect overall performance. You can often achieve better performance by changing configuration settings in CA Single Sign-On Agents, Policy Servers, or the CA Single Sign-On policy objects themselves. The following diagram represents a standard deployment and details the individual components that are central to performance.



1. The types of web and applications servers deployed in your environment can affect how a CA Single Sign-On Agent and Policy Server communicate.

2. The number of available sockets can affect the efficiency in which an agent and Policy Server communicate.

3. CA Single Sign-On policy design can affect the efficiency in which the Policy Server services authentication and authorization requests.

4. The Policy Server performs a series of services to authenticate and authorize users. These services result in number of reads and writes, collectively known as requests, to a user directory. A contributing factor to CA Single Sign-On performance is determining whether your user directories can handle this workload during sustained and peak periods of operation.

5. The user directory itself can affect CA Single Sign-On performance.

Web Tier Performance

This content describes performance tuning for the web tier.

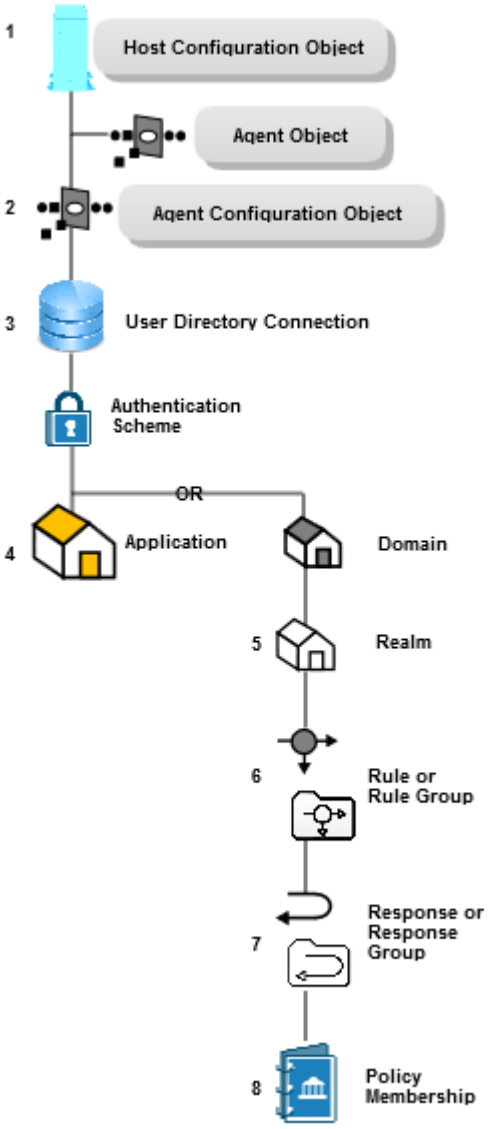
When an agent intercepts a request sent to a web or application server, the agent makes the following calls to the Policy Server:

- isProtected
- isAuthenticated
- isAuthorized

Each of the previous calls generates traffic between the agent in the Web Tier, and the Policy Server in the Application Tier. The following settings can help you adjust the performance of the Web Tier:

- Change the timeout interval for Policy Server requests.
- Change the number of sockets that are available for an agent to use for Policy Server connections.
- Use the agent caches to reduce the number of calls an agent makes to the Policy Server.

The shaded items shown in the following illustration contain settings that affect the performance of your Web Tier:



Server Performance

Aagents can be installed on a number of supported web and application servers. The performance of the hosting server determines the performance of the CA Single Sign-On web tier. The following items affect how your web server performs with CA Single Sign-On:

- The processor speed of your web server
- The amount of memory in your web server

Agent Performance

The following factors influence agent performance:

- Web or application server CPU and available memory.

- Policy Sever latency (how quickly the Policy Server responds to agent requests).

If too few web servers are available to handle the number of requests, the following types of problems can occur:

- Delays of or inability of users to log in.
- Delays in users receiving the resources they requested.
- CPU usage at or near maximum capacity.

Anticipating the number of requests serviced by each web or application server during peak periods can help you determine the ideal number of web servers for your environment.

Use any of the following methods to estimate the number of requests:

- Complete a capacity planning effort.
- Generate a CA Single Sign-On Activity report for each agent in your environment.
- Generate a performance report for your web server.
  For more information, see the documentation provided by your web server vendor.

Web Tier Socket Usage

When an agent starts, it opens the number of sockets specified by the MinSocketsPerPort parameter in the Host Configuration Object on the Policy Server. If more requests are received, the Agent adds a specified number of new sockets to the connection pool until the maximum number of sockets is reached. When all sockets used, any additional requests (up to 300) are held in a queue, until one of the following events occurs:

- A socket pair becomes available and the request is sent to the Policy Server.
- The request times out, and the user must try again to access the resource.

The Host Configuration Object on the Policy Server contains the parameters that control the number of sockets used.

Increase Request Timeout Interval during Heavy Loads

Consider increasing the length of time that requests from agents are held in the Policy Server queue if your network has any of the following conditions:

- Heavy traffic
- Slow connections

The RequestTimeout parameter in the Host Configuration Object on the Policy Server controls how long the Agents wait for responses from the Policy Server. If the interval is too short, the requests time out and the user receives an error message.

Increase the Amount of Available Sockets for the Agent

If your capacity planning estimates reveal that the number of user requests per agent exceeds 60 at any given moment (20 requests in process and 40 in the queue), increase the value of the **MaxSocketsPerPort** parameter.

After increasing the value of the MaxSocketsPerPort parameter in the Administrative UI, verify that the Max Connections setting in the Policy Server Management Console is high enough to accommodate all the Agent processes in your environment. This setting determines the maximum number of connections available to a specific Policy Server.

Note: For multiprocess web servers (such as an Apache-based server in pre-fork mode), you can reduce this number of sockets to one. Because each process uses only a single thread to communicate with the Policy Server, only one socket is required.

Increase NewSocketStep Setting

When the agent requires additional sockets from the connection pool during peak loads, the NewSocketStep parameter determines the number of sockets obtained each time.

If the value of the NewSocketStep parameter is set too low, response time during peak periods suffers because the Agent takes extra time to create socket connections.

To help avoid slow response times, use your capacity planning estimates to determine how many requests your Agents handle, and then increase the value of the NewSocketStep parameter accordingly.

The ideal number for this parameter is one large enough to prevent the Agent from spending too much time creating sockets for requests as the load on the web or application server increases.

We recommend experimenting with different settings until you find what works best in your environment.

Note: For multiprocess web servers (such as an Apache-based server in pre-fork mode), you can reduce this number of sockets to one. Because each process uses only a single thread to communicate with the Policy Server, only one socket is required.

Minimum Sockets per Port Setting

When an agent starts, it opens the number of sockets specified by the MinSocketsPerPort parameter in the Host Configuration Object on the Policy Server. These sockets maintain a constant connection to the Policy Server.

For most types of web and application servers (including Apache-based servers in worker mode), we recommend leaving this parameter at its default setting. Increasing this parameter occupies additional sockets unnecessarily by leaving them open even when the Agent is not receiving any requests for resources.

Note: For multiprocess web servers (such as an Apache-based server in pre-fork mode), you can reduce this number of sockets to one. Because each process uses only a single thread to communicate with the Policy Server, only one socket is required.

Examples of Relationships between Socket Settings

The type web server that you are using determines the relationship between the socket-allocation parameters on the Policy Server.

Since single-process multiple-threaded web servers operate differently than multiple-process single-threaded web servers, the allocation of sockets on your Policy Servers differs for each type of web server.

See the documentation from the vendor of your web server to determine its type.

The following illustration describes the formula for single-process, multiple-threaded web servers:

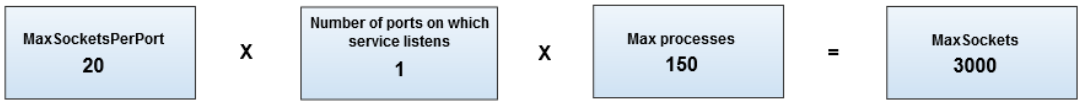
Socket Settings Formula for Single-Process/Multiple-Threaded Web Servers

The following illustration describes the formula for multiple-process, single-threaded web servers:

Socket Settings Formula for Multiple-Process/Single-Threaded Web Servers

The following illustration describes the formula for multiple-process, multiple-threaded web servers:

Socket Settings Formula for Multiple-Process/Multiple-Threaded Web Servers



Use any of the previous formulas as guides when adjusting your socket settings.

Reduce Traffic between Your Agents and the Policy Server

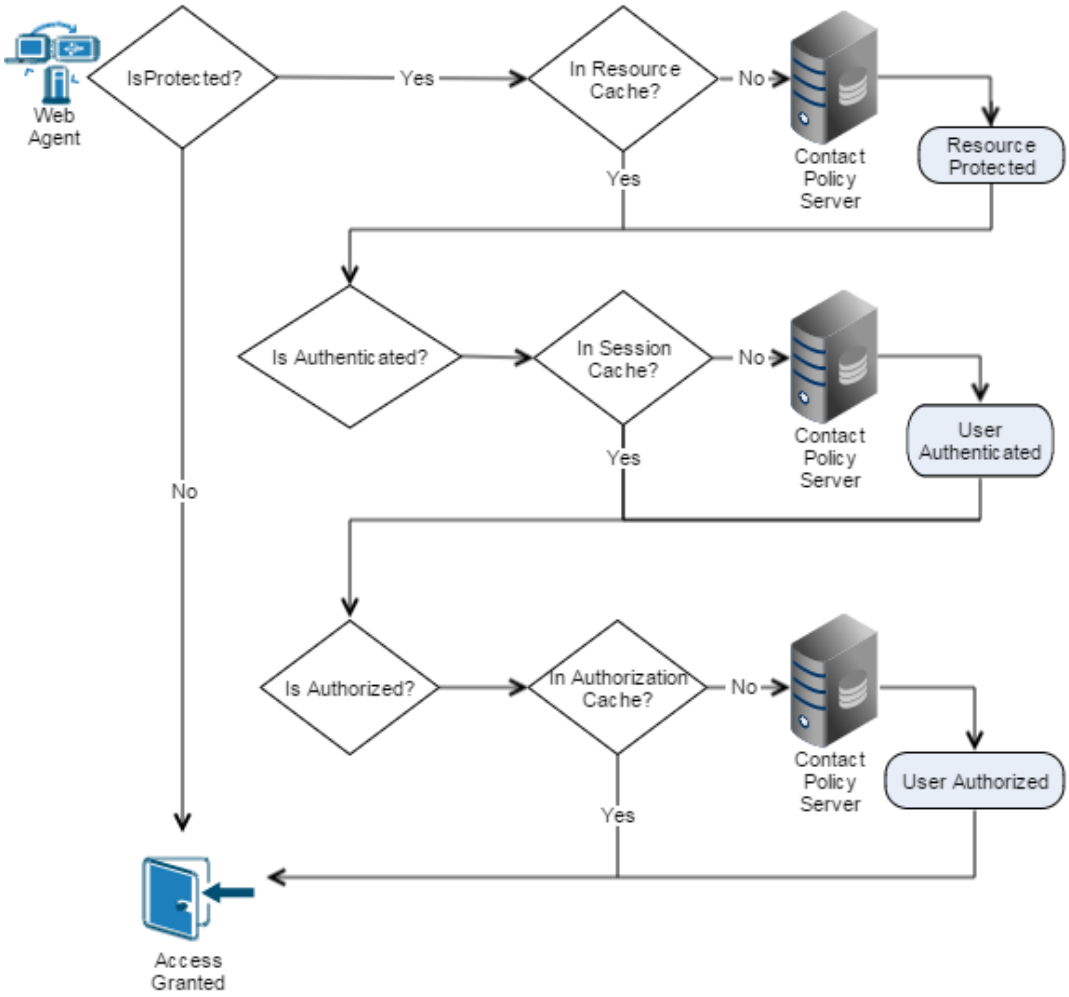
Agents multiple caches and configuration parameters that you can use together to reduce the amount of traffic between your Agents and Policy Servers. Generally, these settings are most efficient in environments where the policies and URIs usually remain static.

How Agent Caches Work

The CA Single Sign-On Agent searches the following caches for the information it needs before contacting the Policy Server:

- Resource Cache
- Session Cache
- Authorization Cache

Because retrieving information from a cache is quicker than contacting the Policy Server, performance improves. The following illustration describes this process:



Resource Cache

Each Agent uses a resource cache to store the following information it receives from the Policy Server temporarily:

- Whether a resource is protected
- Any additional response attributes included in the policy

The Agent searches the resource cache to determine if a resource is protected before contacting the Policy Server. If the resource exists in the cache, traffic to the Policy Server is reduced because the Agent does not make an IsProtected call to the Policy Server.

Two Agent configuration parameters affect the resource cache. Consider the following as you plan your CA Single Sign-On deployment:

- **Resource Cache Timeout**

We recommend basing the timeout interval of the Agent resource cache on the results of your capacity planning tests. A timeout interval that is too small limits the effectiveness of the resource cache. The value of the ResourceCacheTimeout parameter in your Agent configuration determines the timeout interval of the resource cache.

- **Resource Cache Timeout**

  We recommend basing the timeout interval of the Agent resource cache on the results of your capacity planning tests. A timeout interval that is too small limits the effectiveness of the resource cache. The value of the ResourceCacheTimeout parameter in your Agent configuration determines the timeout interval of the resource cache.

Resource Cache and URL Query Strings

If you want to protect applications that use URL query strings, you can still take advantage of the resource cache by configuring the Web Agent to ignore the data in the query string. When the query string data is ignored, the truncated URL is stored in the resource cache. Query strings are ignored by setting the value of the IgnoreQueryData parameter in your Web Agent configuration.

Important! Do not enable this setting if you have policies which depend on URL query data.

The following table shows how ignoring the query strings in a URL determines whether the items from resource cache are used, or if the Web Agent contacts the Policy Server instead:

Requested URL with query string	Truncated URLstored in cache	Cached item used	Policy Server Contacted
/exampleapplication/page1.html?user=firstuser	/exampleapplication/page1.html	No	Yes
/exampleapplication/page1.html?user=seconduser		Yes	No
/exampleapplication/page2.html?user=seconduser	/exampleapplication/page2.html	No	Yes

Session Cache (authentication)

Each Agent uses a session cache to store the authentication information of users whom the Policy Server has already authenticated.

The Agent searches the session cache to determine whether a user is authenticated before contacting the Policy Server for authentication. The session cache improves performance by reducing the number of authentication calls to the Policy Server.

The authentication for a user ends when any of the following events occur:

- The user logs out.
- The session associated with a user expires.

- The age of an item in the cache exceeds 60 minutes.

Authentication information is removed from the session cache and discarded.

Authorization Cache

Each Agent uses an authorization cache to store the authorization identification of users whom the Policy Server has already authorized.

The Agent searches the authorization cache to determine whether a user is authorized before contacting the Policy Server for authorization. The authorization cache improves performance by reducing the number of authorization calls to the Policy Server.

The authorization for a user ends when any of the following events occur:

- The user logs out.
- The session associated with a user expires.

The authorization identification is removed from the cache and discarded.

Session and Authorization Cache Settings

A combination of Policy Server settings and agent configuration parameters control the session cache and authorization cache. Use the results of your capacity planning as a guide to determine the best values for the following settings in your CA Single Sign-On deployment:

- **Session Timeouts**
  We recommend setting the session timeouts as follows:
  - Set the maximum session timeout to match the sustained amount of time that the largest number of users are accessing the protected applications.
  - Set the idle session timeout to an interval that meets all the following criteria:
    - Long enough to prevent the user from being logged out while working.
    - Short enough to log the user out automatically when the application is not being used (such as when a user leaves the computer without logging out).

  Policy Server settings determine the timeout intervals.

- **Session Cache Size**
  Base the size of this cache on the number of users that you expect to access a resource for a sustained period during the session timeout interval. Include users who logout and log back in during the session timeout period in your sizing estimate. Do not include users whom you expect to make relatively few requests in your sizing estimate (because these users have a small effect on the session cache and authorization cache). A Web Agent configuration parameter named MaxSessionCacheSize determines the size of both the session cache and the authorization cache.

Caching and Anonymous Users

The anonymous authentication schemes offered by CA Single Sign-On do *not* provide access control to resources that they protect. Anonymous authentication schemes allow the following for unidentified users on your network:

- Track how often a user returns to your sites.
- Track what a particular user does while visiting your sites (such as the pages the user viewed during a visit).
- Display personalized content for a particular user.

When users request resources protected by an anonymous authentication scheme, the Policy Server assigns a Global Unique Identifier (GUID), and stores it in the browser of the associated user. CA Single Sign-On uses this GUID to identify the user.

If you plan to use an anonymous authentication scheme, implementing the following items can improve performance in your CA Single Sign-On environment:

- Separate web servers to handle the anonymous requests.
- Configure the Web Agent on each separate web server to cache the anonymous requests by setting the CacheAnonymous parameter.

Using separate web severs and Web Agents for anonymous users keeps the caches on the other web servers that service requests for the protected resources from being flushed too often.

Other Parameters That Affect Web Agent Performance

The following parameters also affect Web Agent performance:

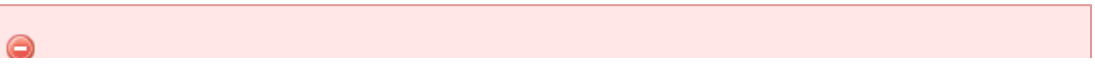
- PSPollInterval
- IgnoreExt
- IgnoreURL

Policy Server Poll Interval Parameter

Agents contact the Policy Server regularly to receive any updated policies or encryption keys. The time interval for contacting the Policy Server can be adjusted by changing the PSPollInterval agent configuration parameter.

Increasing the time interval can reduce unnecessary traffic between the Agents and the Policy Server. Consider increasing the interval when your environment has any of the following characteristics:

- You have many Agents.
- Most of the policies are static, and do not change often.



Important! Increasing the PSPollInterval parameter also affects how quickly the Agents enforce policy changes. For example, suppose you change a Policy to revoke access for a terminated employee at 10:30, and your PSPollInterval parameter has a value of 3600 (the number of seconds in an hour). The Web Agents would not enforce the changed policy until as late as 11:30.

Ignore Extensions Parameter

If the resources you want to protect with CA Single Sign-On contain many images or files that you do *not* want to protect, you can reduce traffic between your Web Agents and Policy Servers by configuring the Web Agent to ignore certain file extensions.

Performance improves because the Web Agent does *not* make the following calls to the Policy Server:

- IsProtected
- IsAuthenticated
- IsAuthorized
- Login

Requests for the associated resources are passed directly to the web server and the user is granted access.

Identifying the resources you want to protect first can help you determine which file extensions, if any, you want your Web Agents to ignore.

Add any file extensions you want to ignore are to the IgnoreExt parameter of your Web Agent configuration.

Ignore URL Parameter

If you want to leave the resources in certain subdirectories unprotected, you can configure the Web Agent to ignore certain uniform resource identifiers (URI).

For example, if each of your web servers has a subdirectory named pictures, and you want to leave those directories on protected, you can set the IgnoreURL parameter in your Web Agent configuration.

Performance improves because the Web Agent does *not* make the following calls to the Policy Server:

- IsProtected
- IsAuthenticated
- IsAuthorized
- Login

Requests for the associated resources are passed directly to the web server and the user is granted access.

Improve Agent Performance through Load Balancing

When you have multiple Agents and Policy Servers, dynamic load balancing reduces latency and improves throughput because the Agents distribute requests among all the Policy Servers. Dynamic load balancing gives the Agents faster access to Policy Servers and more efficient authentication and authorization.

CA Single Sign-On provides software-based failover and load-balancing of their communication with multiple Policy Servers. The EnableFailover parameter of the Host Configuration Object uses one of the following values to determine how Web Agent connections are handled:

- When the value is set to yes, the Agent always tries to connect to the first Policy Server listed (from left to right) in the Host Configuration Object. If you have multiple Policy Servers, all the Agents try to connect to the first one. The other servers in the list are not contacted unless the first server in the list is not available. In high-volume environments, this configuration is less efficient than load balancing because some Policy Servers handle many connections while others handle fewer connections, if any.

- When the value is set to no, load-balancing is enabled. The Agents balance their requests among all the Policy Servers in listed in the Host Configuration Object in a round-robin fashion. We recommend this setting because it produces better throughput when using multiple Policy Servers. Failover still occurs if one of the load balancing Policy Servers is not available.

CA Single Sign-On also supports the use of hardware load balancers to provide high performance dynamic load balancing of connections between Agents and Policy Servers. When configured to expose multiple Policy Servers through a virtual IP address, hardware load balancers handle distribution of load between all Policy Servers associated with that virtual address. Because the Agent does not need to handle failover or load balancing, set the EnableFailover parameter to yes to disable CA Single Sign-On load balancing. Configure only the VIP or VIPs that expose groups of Policy Servers in the Host Configuration Object.

Failover and Load Balancing with Multi-Threaded Web and Application Servers

Agents running on multi-threaded web and application servers (such as Sun Java System, IIS, an Apache-based server in worker mode, or WebSphere Application Server), open the minimum number of sockets to a Policy Server at startup.

If you configure your environment for failover or load-balancing between Policy Servers, then the Agent opens the minimum number of sockets to each Policy Server at startup. Connections to a load-balanced Policy Server occur in the same way, although fewer sockets are opened to each Policy Server, because each is getting only half of the total requests.

If configured for failover, and an error occurs between the Agent and the primary Policy Server, then connections to the failover Policy Server are used. Failover occurs per service, so there could be active connections to both the primary and the failover Policy Servers at once. Once the primary Policy Server comes back up, the sockets opened to the failover server remain. All new sockets are opened to the primary Policy Server.

Failover and Load Balancing with Multi-Process Web and Application Servers

An agent running on a multi-process web or application server (such as an Apache-based server running in pre-fork mode) opens the same number of connections to *all* configured Policy Servers, regardless of whether failover has occurred or not.

When failover occurs, it happens independently for each child, because each child process has its own connections to the Policy Server. This results in a 500 error for each socket as failover takes place. After the primary Policy Server comes back up, the sockets opened to the failover server remain open. All new sockets are opened to the primary Policy Server.

Web Servers, Web Agents, and Web Server Processes

Each agent requires its own web server instance. IIS web servers, for example, operate using a single instance on the computer on which is it installed. The number of IIS agents equals the number of IIS web servers.

For other web servers that support multiple instances per computer, you can install and configure one agent for each instance. For example, you could have one computer that runs three separate web server instances. Each instance has its own agent. Therefore, one computer operates three agents.

For Apache web servers, the following multi-processing modules (MRMs) affect how the agent processes connect to the Policy Server:

- **Pre-fork mode**

  Creates child processes to handle additional requests.

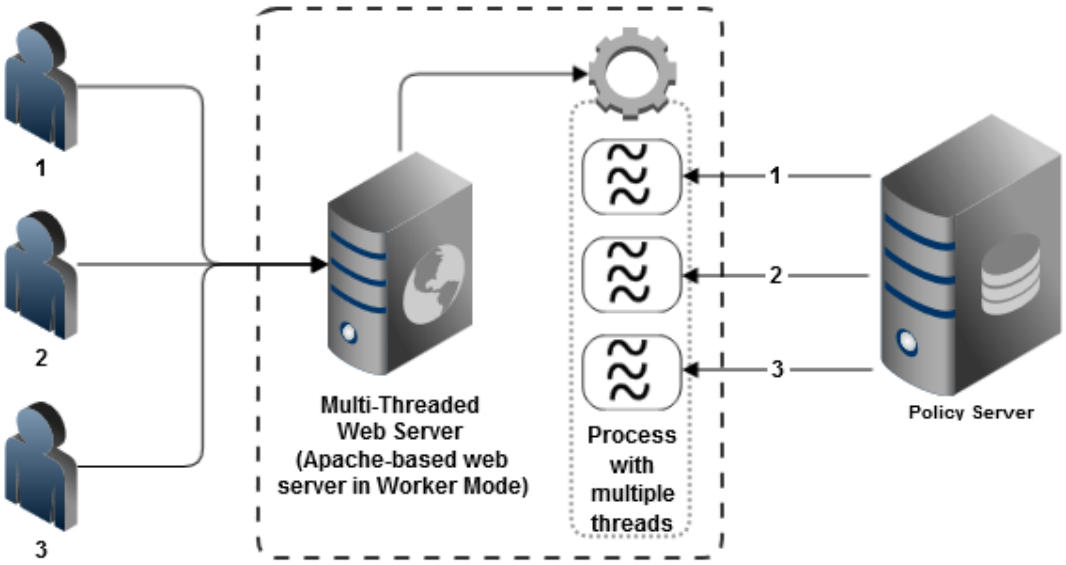
- **Worker mode**

  Obtains additional threads from the connection pool to handle additional requests.

Web Agent and Policy Server Interaction using Apache-based Web Server Worker Mode

Apache-based web servers in worker mode use threads to handle connections to the Policy Server. Threads are obtained from a connection pool as needed to create additional connections to the Policy Server during heavy loads.

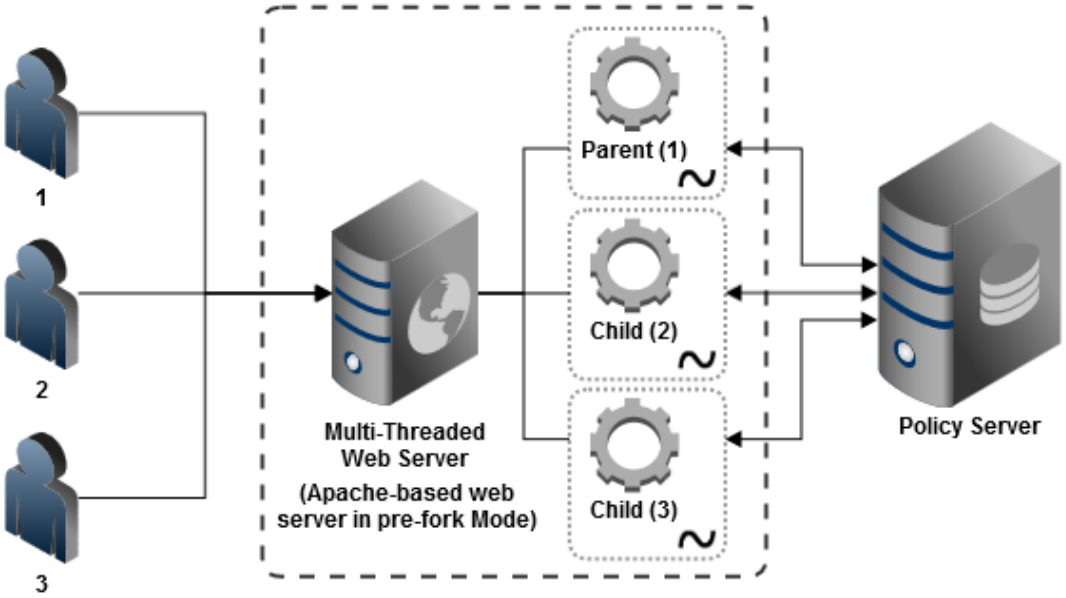
The following illustration describes this process:



Web Agent and Policy Server Interaction using Apache-based Web Server Pre-Fork Mode

When an Apache-based web server in pre-fork mode receives a request, the web server spawns a child process to communicate with the Policy Server. When more requests are received, more child-processes are spawned to handle them. Each child process spawned by the Apache-based web server has its own independent connections to the Policy Server.

The following illustration describes this process:



For Apache-based web servers, the value of the MaxClients parameter (in the httpd.conf file) determines the number of child processes spawned the web server. When a parent process from an Apache-based web server spawns a child process, the child process opens an initial connection to the Policy Server.

An important distinction exists between the number of Web Agents, and the number of Web Agent processes. Each Web Agent requires its own web server instance. IIS web servers, for example, only operate as a single instance, so the number of IIS Web Agents equals the number of IIS web servers. For other types of servers, it is possible to have multiple server instances listening on different ports within one physical web server.

The maximum number of sockets opened from an Apache-based web server to a Policy Server equals the value of the MaxClients parameter multiplied by the number of Web Agent processes. For example, if the value of the MaxClients parameter of your server is set to 150, and you have five Web Agent processes, then the maximum number of possible sockets opened is 750.

Using a multiprocess web server affects the ratio of Web Agent processes to Policy Servers in your CA Single Sign-On environment. The limiting factor often becomes the number of connections between the Web Agent processes and the Policy Server, not the number of transactions per second.

Before deploying Web Agents, verify that the Policy Servers receiving the requests can handle the maximum number of connections that the related web servers could open.

Operating System Tuning for Agents

Contents

Tune the Shared Memory Segments

If you install an Apache-based agent on Solaris systems, tune the shared memory settings of the operating environment for the Agent to function correctly. By increasing the shared memory segments or your operating environment, you improve the performance of the Agent. The variables that control shared memory segments are defined in the specification file of your operating environment.

For AIX operating environments, run the following command before starting an Apache-based server:

```
export EXTSHM=ON
```

Note: Sometimes Linux operating environments require tuning the shared memory segments. For more information about the shared memory segments and how to tune them, see the documentation for your particular operating environment.

Follow these steps:

1. Follow the appropriate procedure for your operating environment:
   - Solaris**:** Open the /etc/system file, using the editor of your choice.
2. Modify the shared memory variables using *one* of the following methods:
   - Solaris: Add the variables shown in the following list and configure them using the recommended settings shown in the examples. Use the following syntax:

```
set shmsys:shminfo_shmmax=33554432
```

- shmsys:shminfo_shmmax

  Specifies the maximum shared memory segment size. Controls the maximum size of the Agent resource and session cache.

  > **Note:** To estimate the amount of memory segments that are required, allocate 4 KBs per entry in each cache, or view cache usage statistics in the OneView Monitor.

  Example: 33554432 (32 MB) for busy sites that require large caches.

- shmsys:shminfo_shmmin

  (Not required for Solaris) Minimum shared memory segment size. Controls the minimum size of the Agent resource and session cache.

- shmsys:shminfo_shmmni

  Specifies the maximum number of shared memory segments that can exist simultaneously, systemwide.

  Example: (except Solaris 9) N/A

  Example: (Solaris 9) 200

- shmsys:shminfo_shmseg

  (Not required for Solaris 9) Specifies the maximum number of shared memory segments per process.

  Example: 24

- semsys:seminfo_semmni

  Specifies the number of semaphore identifiers. Use 11 for every instance of the Agent that you run on the system.

  Example: (except Solaris 9) 100

  Example: (Solaris 9) 200

- semsys:seminfo_semmns

  Specifies the number of semaphores in the system. Use 10 for every instance of the Agent that you run on the system.

  Example: (Solaris 9) 100

  Example: (Solaris 9) 400

- semsys:seminfo_semmnu

  Specifies the number of processes using the undo facility. For optimal performance, set the semmnu value so it exceeds the number of Apache child processes running on the system at any one time. For Apache–based servers, use a value exceeding the maxclients setting by 200 or more.

  Example: (Solaris 9) 200

3. Save your changes then exit the file or the utility.
4. Reboot the system.
5. Verify your changes by entering the command:

   ```
   $ sysdef -i
   ```

How to Tune the Solaris 10 Resource Controls

Tune the resource controls at the project level to improve the performance of the agent.

Note: See your Solaris documentation for more information.

Tuning the resource controls on Solaris 10 uses the following process:

1. Determine the project that is associated with the user account under which the Web Agent runs.
2. Increase the settings for any of the following resource controls of that project:

   - project.max-shm-ids

     Specifies the maximum shared memory IDs for a project.

   - project.max-sem-ids

     Specifies the maximum number of semaphore IDs for a project.

   - project.max-msg-ids

     Specifies the maximum number of message queue IDs for a project.

   - project.max-shm-memory

     Specifies the total amount of shared memory allowed for a project.

   - process.max-sem-nsems

     Specifies the maximum number of semaphores allowed per semaphore set.

- process.max-sem-ops

  Specifies the maximum number of semaphore operations allowed per semop.

- process.max-msg-messages

  Specifies the maximum number of messages on a message queue.

- process.max-msg-qbytes

  Specifies the maximum number of bytes of messages on a message queue.

Application Tier Performance

Contents

Policy Servers evaluate polices in the application tier and user credentials and attributes in the data tier to protect resources. Consider the following guidelines to performance tune the application tier:

- The amount of system resources required to authenticate users affects performance.
- The amount of system resources required to authorize users affects performance.
- The number of Policy Server requests to user directories during authentication and authorization affects performance.

Policy Design and Performance

CA Single Sign-On policies define how users interact with resources. When you create policies in the Administrative UI, you link together (bind) objects that identify users, resources, and actions associated with the resources.

You can improve or degrade performance in the way you configure specific components or by choosing to enable optional features. A performance strategy includes:

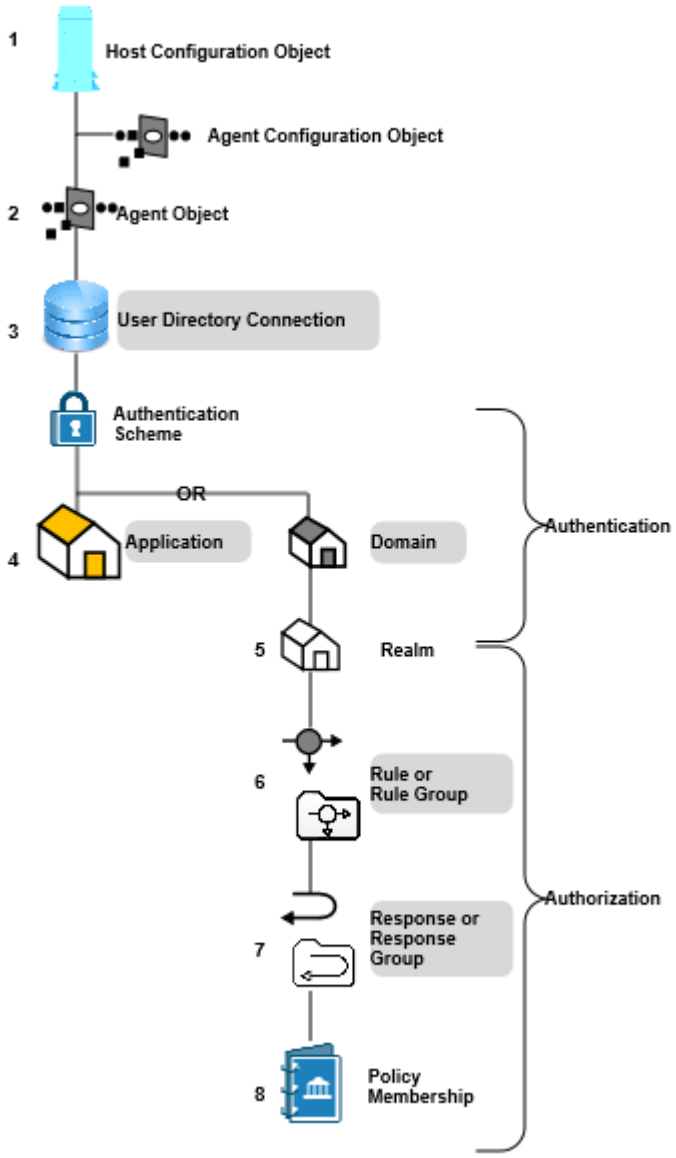
- Identifying the policy objects that can affect performance
- Identifying the parameters and features that affect user authentication
- Identifying the parameters and features that affect user authorization

The business rules and security requirements of your enterprise should ultimately dictate your policy design. The following guidelines are available to help you balance CA Single Sign-On performance, while meeting these requirements.

CA Single Sign-On Policy Objects and Performance Roadmap

CA Single Sign-On requires that you configure core policy objects in a specific order. The following diagram lists this order, where shaded items represents objects that affect performance during user authentication or authorization.

Note: The Host Configuration Object (HCO) and Agent Configuration Object (ACO) affect the performance of your Web tier.



Applications

You can improve or degrade performance during authentication and authorization in the way you configure applications.

An application is a Policy Server object that defines a complete security policy for one or more related web services. Applications associate web service resources with user roles to specify entitlement policies that determine what web service users can access what web service application resources.

When you create an application, you bind it to one or more user directory connections against which the Policy Server attempts to authenticate users. Therefore, the number of directory connections, and order in which they are listed, directly affects CA Single Sign-On performance during authentication.

The number of web service ports and operations that are defined as protected resources in an application correlates to CA Single Sign-On performance during authorization.

Resources can be bound to one or more responses. When a resource is accessed, the associated response returns information to an agent, such as user attributes, DN attributes, static text, or customized active responses.

The types of responses you bind to web service resources directly correlate to CA Single Sign-On performance during authorization.

Domains

You can improve or degrade performance during authentication in the way you configure domains.

A CA Single Sign-On policy domain is a logical grouping of resources associated with one or more user directories. When you create a domain, you bind one or more user directory connections to the domain.

The Policy Server attempts to authenticate users using these directory connections. Therefore, the number of directory connections, and order in which they are listed, directly correlates to CA Single Sign-On performance during authentication.

Realms

You can improve or degrade performance during authentication in the way you configure realms.

You group the resources in a domain into one or more realms. A realm is a set of resources (URLs) with a common security (authentication) requirement. The resource filter you define and the authentication scheme you select directly correlate to performance during authentication:

- The resource filter functions as the root of the protected resources. The Policy Server must evaluate the resource filter to determine if the requested resource is protected (IsProtected?).
- The authentication scheme associated with the realm determines the type of credentials users must present to gain access to the resources in the realm (IsAuthenticated?).

Realm settings also determine:

- How CA Single Sign-On handles user sessions. CA Single Sign-On creates a user session in the context of the realm to which the user authenticated against.
- If the realm can be used to control actions during authentication..

Rules and Rule Groups

You can improve or degrade performance during authorization in the way you configure realms.

You create rules or rule groups in the context of a realm. Rules:

- Identify the specific resources within a realm that require protection
- May be used to either allow or deny access to the resource based on specific authentication or authorization events.

The resource filter you define in the rule, which is prefixed with the realm filter, identifies the resource that requires protection.

The Policy Server evaluates rules to determine which resource filter best matches the requested resource. Upon a match, the Policy Server fires the policies to which the rule is bound to determine if the user is authorized to access the resource.

The number of rules within a realm and how you define each of the resource filters directly correlates to CA Single Sign-On performance during authorization.

Responses

You can improve or degrade performance during authorization in the way you configure responses.

Responses or response groups are bound to specific rule or rule groups. When a rule fires, a response can:

- Customize the amount of time user sessions remain valid.
- Redirect the user to other resources.
- Customize the content the user receives based on attributes contained in a user directory.
- Pass static text, user attributes, DN attributes, customized active responses, or the runtime values of defined variables from the Policy Server to an agent.
- Instruct a WSS Agent to generate WS-Security headers and SAML Session Tickets

Policies rules can be bound to one or more responses. The types of responses you bind to policy rules directly correlates to CA Single Sign-On performance during authorization.

Authentication Guidelines

CA Single Sign-On performance during the authentication (IsAuthenticated?) step typically correlates with:

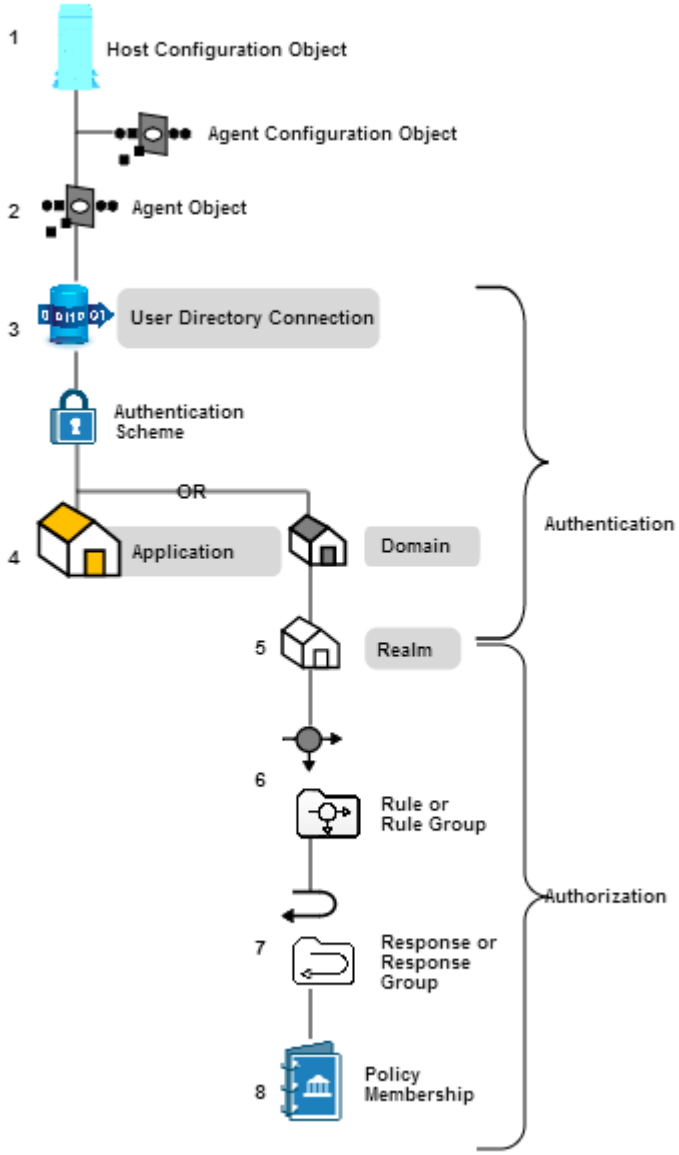
- The system resources used to service an authentication request

- The number of reads/writes, collectively known as requests, that the Policy Server makes to user directories to service an authentication request

Policy Objects and Performance Roadmap

Authentication performance can improve or degrade depending on how you configure specific policy objects or by choosing to enable optional features associated with those objects.

CA Single Sign-On requires that you configure core policy objects in a specific order. The following diagram lists this order, where shaded items represent objects that affect performance during user authentication.



User Directories and Authentication Performance

Configuring a domain requires that you bind one or more user directory connections to the domain. The Policy Server uses the search criteria you specify in the user directory connection to verify user credentials during the authentication step.

The following factors affect user authentication performance at the directory level:

- Search expressions and queries—The more complex the LDAP expression or ODBC query, the longer it takes the Policy Server to resolve the criteria to authenticate the user.
- Password Services—You can apply password policies to user directories. Consider the following before implementing password policies:
  - The Policy Server reads attributes related to the password policy and may need to update them. Updating an attribute requires the Policy Server to write to the user directory.
  - If the password policy is configured to track login details, an additional user directory write is required for every authentication.
  - The Policy Server takes longer to resolve password policies that only apply to a specific group of users within the directory, instead of the entire directory.

CA Single Sign-On Web Services Security Authentication Schemes and Authentication Performance

Different Web Services Security authentication schemes impose different level of WSS Agent processing overhead, which can also vary between WSS Agent types.

In general, authentication throughput is greater for authentication schemes that do not require digital signature verification or payload confidentiality.

Digital signature verification is more CPU- and data-intensive on WSS Agent for Web Servers, but also slightly impacts WSS Agents for application servers.

Domains and Authentication Performance

The following factors affect user authentication performance at the domain (or application object general) level:

- The number of directory connections in the domain—The Policy Server searches each user directory in the domain until it is able to validate the user credentials. The greater the number of user directory connections, the longer it can take the Policy Server to authenticate the user. Evaluate ways to reduce the number of directory connections in a domain to prevent unnecessary Policy Server requests. Consider:
  - Who is requesting the resources within the domain and in which directories their information is stored
  - Combining user directories when you add an organization to your CA Single Sign-On deployment

- The order in which user directory connections are listed—The Policy Server searches user directories in the order in which the domain lists them. Evaluate authentication priorities when determining the order of connections. Consider:
  - If a larger percentage of users access the application from a specific directory or directories
  - If a smaller group of users exists that are higher priority for authentication

Realms and Authentication Performance

The following factors affect user authentication performance at the realm (or application object component) level. Consider each as you configure realms:

- Credential collection—Realms are associated with a specific authentication scheme, some of which require the use of credential collectors. Agents protecting resources with these types of authentication schemes redirect users to the credential collector to gather the credentials. Gathering credentials adds an additional step to the authentication process.
- Persistent Sessions—When CA Single Sign-On authenticates a user, the Policy Server issues a session ticket. A session ticket contains basic information about the user and the authentication context of the user. By default, CA Single Sign-On implements session management through non–persistent sessions, for which the Agent writes the session ticket to a cookie in the web browser of the user.
  Some CA Single Sign-On features require persistent sessions. You can configure a realm for Persistent Sessions. Agents protecting resources in this realm write the session ticket a CA Single Sign-On session store, which results in additional requests to the session store for each authentication.

  Important! Persistent sessions can have a significant impact on performance.

- Authentication Events—By default, a realm is configured to Process Authentication Events. This setting lets you define rules that fire when a user authenticates or fails to authenticate. Policy evaluation logic applies to all realms configured to process Authentication Events. This logic consumes system resources and can result in user directory requests.
  Evaluate the need for event actions that occur when users authenticate to gain access to a resource. If you do not require authentication actions, disable Authentication Events for the realm to speed the authentication step.

Authorization Guidelines

CA Single Sign-On performance during the authorization step typically correlates with:

- The system resources used to service an authorization request.
- The number of reads/writes, collectively known as requests, the Policy Server makes to user directories to service an authorization request.

The complexity of your CA Single Sign-On policy design affects each of these areas.

Policy Objects and Performance

You can improve or degrade authentication performance in the way you configure specific policy objects or by choosing to enable optional features associated with those objects. The following policy objects can affect performance during user authorization:

- Rules (see page 119)
- Responses (see page 120)
- Policy membership (see page)

Rules and Authorization Performance

The following factors affect user authorization performance at the rule (or application object resource) level:

- Large numbers of rules in a single realm can slow authorization decisions. If a user is authenticated for a particular realm, the Policy Server must evaluate all rules within the realm to determine which of the resource filters best matches the specific resource (URL) the user is requesting.
- The type of resource filter affects how quickly the Policy Server can evaluate the resource match.

The following filters are listed in the order in which they have the smallest affect on performance:

Exact match—Defining a resource filter with a specific resource has the smallest affect on performance. The Policy Server only has to compare the resource filter to the URL of the requested resource.

Example: A company creates a customer realm (/customer) and specifies a rule with a specific page of their portal application (lending_home.html). The resulting resource filter is /customer /lending_home.html. Evaluating a match between the requested resource and the rule only requires the Policy Server to compare the requested resource with the resource filter to determine if it is a match.

- Exact prefix—Defining a resource filter with a prefix has a greater affect on performance than an exact match. The Policy Server must determine if the requested resource is contained within the root (realm) of the resource.
  Example: A company creates an employee realm (/employee) and specifies a rule with "*.html". The * prefix specifies that all html files in the employee realm are protected. The resulting resource filter is /employee /*.html. Evaluating a match between the requested resource and the resource filter requires the Policy Server to evaluate if the requested resource is part of the employee directory and is an HTML file.
- Regular expression—Defining a resource filter with a regular expression has the greatest affect on performance. The Policy Server must evaluate the expression and compare the result to the requested resource. The complexity of the expression further affects performance.

Responses and Authorization Performance

The type of response attributes bound to rules in a CA Single Sign-On policy affect performance. The following response types are listed in the order in which they have the smallest affect on performance:

- Static—Defining a static attribute returns data that is constant.
- User attribute—Defining a user attribute returns profile information from a user's entry in a user directory.

  Note: This type of response requires the Policy Server to search the user directory.

- DN attribute—Defining a DN attribute returns information associated with directory objects to which the user is related. Groups to which a user belongs, and organizational units (ou) that are part of a user DN, are examples of directory objects whose attributes can be treated as DN attributes.

  Note: This type of response requires the Policy Server to search the user directory.

Policy Membership and Authorization Performance

Policy membership is the part of a CA Single Sign-On policy that specifies which users apply to the policy. Policies are stored in domains, and as a result, you use filters to apply policy membership to any or all users stored in the user directories bound to the domain. The type of filter you define determines how the Policy Server evaluates policy membership.

The following filters are listed in the order in which they have the smallest affect on performance:

- All—"All" has the smallest affect on performance.
  When CA Single Sign-On authenticates a user, the Policy Server issues a session ticket. The session ticket identifies the user directory in which the user is stored. The Policy Server only has to compare the session ticket with the directory bound to the policy to determine that the policy applies to the user.
- Distinguished name—A distinguished name (dn) has a greater affect on performance than "All". The organization or organizational unit, which contains the dn of the authenticated user, is stored in the session ticket. The Policy Server has to compare the session ticket information with the policy membership filter to determine if the policy applies to the user.
- Group membership or search expressions—These types of filters have a greater affect on performance than distinguished names. Group membership and search expressions consume additional system resources and result in a user directory search. The Policy Server must:

  1. Resolve the group membership or search expression
  2. Search the user directory to determine if the policy applies to the user.

- Nested groups—Defining policy membership with a nested group has the greatest affect on performance.
  The Policy Server must search each user group and all sub–groups in the directory to determine if the policy applies to the user.

Important! Directories with deep group hierarchies can have a significant effect on the time it takes the Policy Server to evaluate policy membership.

Note: You can enable the User Authorization cache to reduce the number of requests the Policy Server makes to user directories to resolve policy membership.

User Authorization Cache

The user authorization cache reduces the number of user directory requests to determine policy membership by storing the relationship between users and policies.

Note: The user authorization cache does not store data about the user, store user attribute values, or cache user entries.

For example, three policies are configured to apply to an "Administrator" group, to which user A belongs. The first–time the Policy Server evaluates policy membership, it must resolve the group membership and make three requests (one for each policy) to the user directory to determine that each policy applies.

The Policy Server writes these results to the user authorization cache. Subsequent policy evaluation does not require the Policy Server to make user directory requests. Rather, the Policy Server uses the cached authorization information to determine policy membership.

Note: The Policy Server polls for policy updates periodically. The default interval is 60 seconds. If the policy membership changes, the Policy Server reloads the policy and removes the cache entries that are related to the updated policy.

User Authorization Cache Efficiency

The user authorization cache is most efficient when:

- All user requests during a session are consistently sent (persisted) to the same server.
- All agents are configured for Policy Server failover, not round–robin load balancing.

If these factors are not met, the efficiency of the User Authorization cache is reduced.

Example: the user authorization cache and agents configured to round–robin load balance

The more Policy Servers that are in the agent round–robin pool, the greater the chance that the efficiency of the user authorization cache is reduced.

If a single agent is configured to round–robin between two Policy Servers, the first request for a protected resource results in a user authorization cache entry on one of the Policy Servers. There is approximately a 50 percent chance that the Policy Server that does not have the cache entry must service the second request. Moving forward, however, both Policy Servers have cached the data for subsequent requests.

Consider now, the effect of a single Agent configured to round–robin between 10 Policy Servers. After a Policy Server authorizes a user and enters the result in to the authorization cache, there is only a 10 percent chance that the same Policy Server services the next request. In this configuration, 5 cache misses must occur before there is a 50 percent chance of a cache hit.

Note: Policy Server clusters can reduce the effect round–robin load balancing has on the user authorization cache.

Estimate the Size of the User Authorization Cache

The default size of the user authorization cache is 10 MB. You can estimate the amount of space the user authorization cache requires and use the Policy Server Management Console to adjust the default size.

To estimate the size of the user authorization cache

1. Use the following formula to estimate the number of cache entries:
   *expected_users * number_of_policies_per_session = entries*

   - **expected_users**
     Specifies the total number of users authenticating to the applications CA Single Sign-On is protecting.

   - **number_of_policies_per_session**
     Specifies the average number of policies that apply to a user during the session.

     Note: Each policy has the potential to enter a unique entry into the user authorization cache.

   - **entries**
     Specifies the number of cache entries authorizations can create.

2. Use the following formula to estimate the size of the cache:
   (*entries* * .000062) + 1

   Note: .000062 represents the approximate size of a cache entry in MB.

Auditing and Performance

By default, the Policy Server writes audit events to a text file, which is known as the Policy Server log. Optionally, you can configure the Policy Server to log events to an audit database.

Consider the following factors if you decide to log events to an audit database:

- Performance associated with authentication and authorization is affected because CA Single Sign-On is logging all authentication and authorization decisions to the database.
- (Optional) Synchronous logging—You can configure synchronous logging at the realm level. If configured, the Policy Server prevents the result of each authentication and authorization request until the record is saved in the audit database. Users are not authenticated or authorized until the record is saved.

Load Balancing the Application Tier

Tuning the various agent parameters and following the CA Single Sign-On policy design guidelines may not significantly improve the amount of time it takes the Policy Server to service authentication and authorization requests.

When you have multiple Agents and Policy Servers, dynamic load balancing reduces latency and improves throughput because the Agents distribute requests among all of the Policy Servers.

Data Tier Performance

Contents

Poor performance associated with CA Single Sign-On data stores, especially user directories, is one of the most common reasons for poor CA Single Sign-On performance. Data tier performance typically correlates with two general areas:

- The data tier itself. A user directory that is not properly tuned or lacks sufficient system resources can degrade CA Single Sign-On performance.
- The capacity under which your user directories have to operate. CA Single Sign-On authentication and authorization services result in a number of reads and writes, collectively known as requests, to a user directory. Conduct a capacity planning effort on the user directory itself to be sure it can handle the CA Single Sign-On workload.

A performance strategy includes:

- Determining that the data tier itself is not the primary reason for poor performance.

- Identifying the number of authentications and authorizations CA Single Sign-On must service in a given period.

  Note: The sustained and peak rates at which user authentication and authorization occur can be calculated.

- Estimating how many user directory requests each user authentication, and the subsequent authorizations, create.

Data Tier Guidelines

The Policy Server interacts with the data tier using standard protocols. If your directory servers and databases are tuned to maximize performance with their normal clients, then these modifications can translate into improved CA Single Sign-On performance.

Note: See your vendor-specific documentation for tuning guidance.

There are several general considerations to improving CA Single Sign-On performance as it relates to the performance of your user directories. Examine the following areas:

- The system resources available to the user directory and any external resources that may contend for those resources
- The use of Secure Socket Layer
- The efficiency in which CA Single Sign-On can search the user directory
- The use of static IP addresses
- The use of replication

System Resources

The system resources available to the user directory directly correlates to CA Single Sign-On performance. If the user directory is operating at a high level of utilization, then no amount of CA Single Sign-On tuning can improve performance.

Be sure that the system hosting the user directory is not degrading performance due to:

- A slow CPU or I/O system
- Insufficient memory
- An incorrectly configured buffer cache
- Insufficient or fragmented disk space

Secure Socket Layer and User Directories

Consider the following if you are planning to implement SSL in your CA Single Sign-On environment:

- Configuring the Policy Server and an LDAP user directory to communicate over SSL reduces performance. Review your security requirements to determine if SSL is mandatory.
- If you decide to configure SSL, do not place an SSL accelerator between the Policy Server and the directory server or the Policy Server assumes a single instance of the directory. This can cause inconsistent writes across multiple user directories behind the accelerator.

Static IP Addresses and User Directories

When you configure user directory connections in the Administrative UI, consider using static IP addresses rather than hostnames. Although the time the Policy Server takes to resolve hostnames is negligible, using static IP addresses removes Domain Naming Services (DNS) dependencies.

User Directory Searches

Making sure that CA Single Sign-On can efficiently search users directories directly correlates with performance. Consider the following:

- Use directory indexing to enhance search results for CA Single Sign-On:
  - LDAP—the objectClass attribute, in addition to all other attributes used in searches, should be indexed.

    Note: Microsoft recommends using the objectCategory attribute instead of objectClass. Failing to index the objectClass attribute in Active Directory can result in significant performance degradation.

  - ODBC—All fields defined as search criteria in CA Single Sign-On schema queries should be indexed.

    Note: See your vendor-specific documentation for more information about indexing.

- Design queries to return manageable sets of user groups.

  Note: If you are unable to optimize the query, set the maximum search results parameter to limit large result sets from degrading overall performance.

- Optimize SQL query schemes for ODBC with any standard SQL analyzer.

Replication

Replication can degrade performance in the following situations:

- When the master replica, in a master–slave replication, only allows write requests. Password Services usually requires updates to the password blob attribute for each authentication. If only the master replica can process writes, then each write request is redirected to the master.
  The redirection results in additional time spent on the authentication step, and the master-replica may not be able to accommodate the rate at which writes occur.

- When LDAP referrals are enabled. LDAP referrals can degrade performance because each request may involve more than one request to a directory.

User Store Capacity Planning

The Policy Server performs a series of services to authenticate and authorize users. These services result in number of reads and writes, collectively known as requests, to a user directory. A significant contributing factor to CA Single Sign-On performance is determining whether your user directories can handle this workload during sustained and peak periods of operation.

The following general factors influence CA Single Sign-On performance:

- Total operations and sustained user directory search rates—Total operations is the combined number of requests the Policy Server must service when handling authentication and authorization requests. The rate at which these operations occur fluctuate throughout your business day.
  In turn, the rate at which the Policy Server makes user directory requests to process the operations fluctuates. Some periods generate relatively few user directory requests, while others generate more.
  The sustained user directory search rate represents a period during which the Policy Server makes an average number of user directory requests to service an average number of operations.

- Total operations and peak user directory search rates—During sustained periods of activity, user activity can spike. The peak user directory search rate represents a period during which the Policy Server makes the highest number of user directory requests to process peak numbers of operations.

We recommend using the following guidelines to estimate the load under which your user directories have to operate. Once you have estimated the load, you can use any standard tool to create the load on the directory and track the results.

Note: Many factors can contribute to failing to achieve the required numbers. See your vendor–specific documentation for tuning guidance.

User Store Capacity Planning Checklist

Estimating the number of user directory requests that the Policy Server must make to service authentication and authorization requests requires specific information. Gather the following before beginning a user store capacity plan:

- The total number of daily authentications (authentication load) for the application.

- The total number of daily authorizations (authorization load) for the application.

- The sustained and peak periods during which users are authenticating to the application and requesting protected resources.

Note: A capacity planning effort can help you identify metrics related to authentication load, authorization load, and sustained and peak levels of user activity.

- The total number of enabled policies. For each policy determine:
  - If the policy membership filter results in one or more user directory searches.
  - If the responses bound to the policy results in one or more user directory searches.

How to Estimate a Sustained User Directory Search Rate

Estimating a sustained user directory search rate is the process of determining:

- How the total number of user directory requests fluctuate throughout your business day
- How the user directory requests translate into requests per second over a sustained period.

Complete the following steps to estimate the sustained user directory search rate:

1. Use the authentication guidelines to estimate the number of user directory requests that the authentication load creates.
2. Use the authorization guidelines to estimate the number of user directory requests that the authorization load creates.
3. Estimate the sustained user directory search rate.

Use Authentication Guidelines to Estimate Directory Searches

A Policy Server makes a number of user directory requests to service each authentication request. Some of the user directory requests are required, while others can be avoided.

Estimate the number of Policy Server requests that each authentication creates using the following guidelines:

(Required) Two searches to authenticate each user:

- One search/query, per store, to identify the user
- One search/query to verify the user credentials

(Optional) Additional searches may be required depending on how you design policies and if you decide to enable Password Services:

- One search/query for each policy that is bound to a rule that fires when a user is authenticated (OnAuth rule).
- One search/query for each policy that is bound to a response that returns user attributes.
- One write/update per user store enabled for Password Services. If Password Services does not apply to the user directories in the policy domain, a write/update is not required.

The following use cases detail how you can use each guideline to determine the total number of user directory searches the authentication load creates.

Case 1 User Authentication and Directory Requests

A company has:

- Deployed one user directory for their banking application.
- Completed a capacity planning effort. The results of which show that users create an authentication load of 88,000 logins.

The company uses the following formula to begin estimating the number of requests the Policy Server sends to the user directory to service the authentication load:

*authentication_load * 2 * number_of_user_stores = requests_for_authentication*

- **authentication_load**
  Specifies the number of daily authentications for the application.

  Note: Two (2) is a constant. Authenticating a users results in two requests. One search to identify the user and one bind to verify credentials.

- **number_of_user_stores**
  Specifies the number of user stores in the implementation.
- **requests_for_authentication**
  Specifies the number of user directory requests that the authentication load creates.

Result: 88,000 * 2 * 1 = 176,000 requests.

The company uses this estimate to determine the total number of user directory requests required to service the daily authentication load.

Case 2 Policy Design and User Directory Requests

A company has configured four policies to protect the application portal, one of which is bound to a rule that fires upon a successful authentication.

The company uses the following formula to continue estimating the number of requests the Policy Server sends to the user directory to service the authentication load:

authentication_load * (*percent_of_policies * number_of_searches*) = *requests_for_authentication*

- **authentication_load**
  Specifies the number of daily authentications for the application.
- **percent_of_policies**
  Specifies the total number of enabled policies, represented as a percentage, that are:
  - bound to an onAuth rule

- create the same number of user directory searches

Example: Four enabled policies exist. One is bound to an OnAuth rule. This policy generates one user directory search to determine policy membership. Twenty–five percent of the enabled policies fire on authentication and generate one user store search. The remaining policies do not fire during authentication.

- **number_of_searches**
  Specifies the number of requests that the Policy Server makes to determine if the policy applies to each authenticated user.

- **requests_for_authentication**
  Specifies the number of user directory requests that the authentication load creates.

Result: 88,000 * 0.25 * 1 = 22,000 requests

The company uses this estimate to determine the total number of user directory requests required to service the daily authentication load.

Case 3 Responses and User Directory Requests

A company has defined one policy with an OnAuth rule. This policy requires that a common name (cn) attribute response be returned when the policy fires. The company defines a Web Agent response to return this value and binds it to the policy rule.

The company uses the following formula to continue estimating the number of requests the Policy Server sends to the user directory to service the authentication load:

*authentication_load * percent_of_policies * number_of_responses_per_policy = requests_for_authentication*

- **authentication_load**
  Specifies the number of daily authentications for the application.

- **percent_of_policies**
  Specifies the total number of enabled policies, represented as a percentage, that are bound to a specific number of responses that return user attributes.
  Example: If there are four enabled policies, and one uses a response to return a user attribute, then twenty–five percent of the policies require a user directory search.

- **number_of_responses_per_policy**
  Specifies the number of responses bound to the policy.

- **requests_for_authentication**
  Specifies the number of user directory requests that the authentication load creates.

Result: 88,000 * 0.25 * 1 = 22,000 requests

The company uses this estimate to determine the total number of user directory requests required to service the daily authentication load.

Case 4 Password Services and Directory Requests

A company has enabled Password Services for their user store. The company uses the following formula to continue estimating the number of requests the Policy Server sends to the user directory to service the authentication load:

authentication_load * 1 = *requests_for_authentication*

- **authentication_load**
  Represents the number of daily authentications for the application.

  Note: One (1) is a constant. Tracking user login details requires one write to the user directory for each authentication.

- **requests_for_authentication**
  Represents the number of user directory requests that the authentication load creates.

Result: 88,000 * 1 = 88,000 requests.

The company uses this estimate to determine the total number of user directory requests required to service the daily authentication load.

Case 5 Total Directory Requests for Authentication

A company uses the individual totals from each use case to determine the total number of requests the Policy Server sends to the user store to service the authentication load:

- 176,000 requests to identify 88,000 unique users and their credentials
- 22,000 requests to determine if the OnAuth policy applies to those users
- 22,000 requests to return the common name attribute upon authentication
- 88,000 requests for the password policy

Result: 176,000 + 22,000 + 22,000 + 88,000 = 322,080 requests

The company uses this result and the results based on the authorization load to estimate the sustained rate at which the user store must service Policy Server requests.

Use Authorization Guidelines to Estimate Directory Searches

A Policy Server makes a number of user directory requests to authorize a user. Some of the user directory requests are required to determine policy membership, while others are dependent on policy design. You can estimate the number of Policy Server requests that each authorization creates using the following guidelines.

- One search/query for each policy in the policy domain.

Note: This guideline only applies to policies whose membership filter results in one or more user directory requests. For more information about the relationship between policy membership and user directory requests, see Policy Membership and Authorization Requests.

- One search/query for each policy that is bound to a response that returns user attributes.

Note: For more information about the relationship between responses and user directory requests, see Responses and Authorization Performance.

The following use cases detail how you can use each guideline to determine the total number of user directory searches the authorization load creates.

Note: The user authorization cache can significantly reduce the number of authorization-related requests to user directories.

Case 1 Policy Membership and User Directory Requests

A company has enabled three policies protect their portal application:

- Policy A requires one user directory request to determine policy membership.
- Policies B may require up to two user directory requests to determine policy membership.
- Policies C may require up to three user directory requests to determine policy membership.

Additionally, the results of a capacity planning effort show that the application has an authorization load of 726,000.

The company uses the following formula to begin estimating the number of requests that the Policy Server sends to the user directory to service the authorization load:

authorization_load x *percent_of_policies* * *number_of_searches* = *daily_authorization_requests*

- **authorization_load**
  Specifies the number of daily authorizations for the application.
- **percent_of_policies**
  Specifies the number of enabled policies, represented as a percentage, that may result in the same number of user directory requests to determine policy membership.

Note: The total percentage must equal 100 percent.

- **number_of_searches**
  Specifies the number of user directory requests that the Policy Server may make to determine policy membership.

- **daily_authorization_requests**
  Specifies the number of user directory requests to service the authorization request.

Result:

- Policy A—792,000 * 0.33 * 1= 261,360 requests
- Policies B and C—792,000 * 0.66 * 2= 1,045,440 requests
- Total user directory requests-158,000 + 1,045,440= 1,306,880 requests

The company uses this estimate to determine the total number of user directory requests required to service the daily authorization load.

Case 2 Responses and User Directory Searches

A company has enabled three policies to protect their portal application, two of which are bound to responses that return user attributes:

Policy A returns one user attribute when it fires.

- Policy B returns two user attributes when it fires.
- Policies C is not bound to responses that return user attributes.

The company uses the following to estimate the number of user directory requests that the Policy Server makes to resolve responses that return user attributes:

*authorization_load * percent_of_policies * number_of_responses= daily_authorization_requests*

- **authorization_load**
  Specifies the number of daily authorizations for the application.
- **percent_of_policies**
  Specifies the number of enabled policies, represented as a percentage, that result in the same number of user directory requests because of responses returning user attributes.

  Note: The total percentage must equal 100 percent.
- **number_of_responses**
  Specifies the number of responses bound to the policy.
- **daily_authorization_requests**
  Specifies the number of user directory requests to service the authorization request.

Result:

- Policy A—792,000 * 0.2 x 1= 158,000
- Policy B—792,000 * 0.2 x 2= 316,800

- Policies C—792,000 * 0.6 x 0= 0
- Total user directory request—158,000 + 316,800 + 0= 526,000

The company uses this estimate to determine the total number of user directory requests required to service the daily authorization load.

Case 3 Total Directory Requests for Authorization

The company uses the individual totals from each use case to determine the total number of requests the Policy Server sends to the user directory to service the authorization load:

- 1,203,440 requests to resolve policy membership.
- 526,000 requests to return user attributes associated with responses.

Result: 1,203,440 + 526,000= 1,729,440 requests

The company uses these result and the results based on the authentication load to estimate the sustained rate at which the user store must service Policy Server requests.

Estimate the Sustained User Directory Search Rate

The sustained user directory search rate is based on the total number of operations (authentication load plus authorization load), specifically, when and at what rate these requests occur. The chance that these requests are uniformly spread across your business day is unlikely. Rather, the rate at which these requests occur fluctuates, remaining between the lowest and highest (peak) levels for a sustained period.

Estimating the sustained user directory search rate is the process of identifying:

- A sustained period during which the system is servicing an average number of operations.
- How these requests translate into user directory searches.

When estimating the sustained user directory search rate, we recommend using the daily authentication load and authorization load to identify:

- The rate at which total operations occur throughout the day

  Note: We recommend beginning with an evaluation period of 24 hours, broken down into one-hour increments. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

- The sustained period during which the system is servicing an average number of requests
- The approximate number of requests that occur during the sustained period.

Case Estimate the Sustained User Directory Search Rate

The company has determined that:

- The daily authentication load and authorization load for the application result in approximately 888,000 total operations.
- The total operations result in approximately 2,051,520 user directory requests.
- The system is operating at sustained levels for approximately five hours (9:00 AM - 2:00 PM).
- During sustained levels, approximately 84,000 operations occur, per hour.
- Approximately 420,000 (84,000 * 5) operations, or 48 percent (420,000 / 880,000) of the total operations, occur during these hours.

The company uses the following formula to estimate the sustained user store search rate:

(*total_user_directory_requests* * *percentage_of_requests*) / *number_of_hours* / 3600 = *sustained_user_directory_search_rate*

- **total_user_directory_requests**
  Represents the daily number of requests the Policy Server makes to the user directory to service authentication and authorization requests.
- **percentage_of_requests**
  Represents the percentage of total operations that occur when the system is operating at sustained levels.
- **number_of_hours**
  Represents the number of hours when the system is operating at a sustained rate.
- **sustained_user_directory_search_rate**
  Represents the number of requests, per second, the Policy Server makes to the user directory to maintain the sustained rate of operation.

Result: (2,051,520 * 0.48) / 5 /3600 = 54.7 user directory requests per second.

The Policy Server makes 54.7 requests, per second, to the user directory when servicing authentication and authorization requests during sustained levels of operation.

Estimate the Peak User Directory Search Rate

The peak user directory search rate is based on the total number of operations (authentication load plus authorization load), specifically, when and at what rate the system is operating at peak levels. Estimating the peak user directory search rate is the process of identifying when the system is servicing the highest level of operations and how these requests translate into user directory searches.

When estimating the peak authorization rate, we recommend using the metrics that you gathered when determining the sustained authorization rate to determine:

- The hour the system is servicing the highest number operations.
- The approximate number of operations that occur during this period.

Case Estimate the Peak User Directory Search Rate

A company has determined the application results in a total of 888,000 operations per day. These operations result in approximately 2,051,520 user directory searches. Using metrics gathered during a capacity planning exercise, the company has determined that during the single busiest hour, approximately 278,000 operations, or 31 percent of the total operations, occurred.

The company uses the following formula to estimate the peak user store search rate.

(*total_user_directory_requests* * *percentage_of_requests*) / *number_of_hours* / 3600 = *peak_authentication_request_rate*

- **total_authentication_requests**
  Represents the total number of requests the Policy Server sends to the user store.
- **percentage_of_requests**
  Represents the percentage of operations that occur when the system is operating at peak levels.
- **number_of_hours**
  Represents the number of hours in which the system operates at peak levels.
- **peak_user_directory_request_rate**
  Represents the number of requests, per second, that the Policy Server makes to the user store to maintain the peak authentication rate.

Result: (2,051,520 * 0.31) / 1 / 3600 = 176.6 requests per second.

The Policy Server makes 176.6 requests, per second, to the user directory when servicing authentication and authorization requests during peak levels of operation.

User Store Capacity Planning

The Policy Server performs a series of services to authenticate and authorize request messages. These services result in number of reads and writes, collectively known as requests, to a user directory. A significant contributing factor to CA Single Sign-On performance is determining whether your user directories can handle this workload during sustained and peak periods of operation.

The following general factors influence CA Single Sign-On performance:

- Total requests and sustained user directory search rates—The Policy Server must handle an authentication and authorization operation for each incoming request. The rate at which these requests occur fluctuate throughout your business day.
  In turn, the rate at which the Policy Server makes user directory requests to process the operations fluctuates. Some periods generate relatively few user directory requests, while others generate more.
  The sustained user directory search rate represents a period during which the Policy Server makes an average number of user directory requests to service an average number of operations.
- Total requests and peak user directory search rates—During sustained periods of activity, request activity can spike. The peak user directory search rate represents a period during which the Policy Server makes the highest number of user directory requests to process peak numbers of authentication and authorization operations.

We recommend using the following guidelines to estimate the load under which your user directories have to operate. Once you have estimated the load, you can use any standard tool to create the load on the directory and track the results.

Note: Many factors can contribute to failing to achieve the required numbers. See your vendor–specific documentation for tuning guidance.

User Store Capacity Planning Checklist

Estimating the number of user directory requests that the Policy Server must make to service requests requires specific information. Gather the following before beginning a user store capacity plan:

- The total number of daily requests (request load) for the resource.
- The sustained and peak periods during which users are sending requests to the resource.
- The total number of enabled policies. For each CA Single Sign-On policy determine:
  - If the policy membership filter results in one or more user directory searches.
  - If the responses bound to the policy result in one or more user directory searches.

How to Estimate a Sustained User Directory Search Rate

Estimating a sustained user directory search rate is the process of determining:

- How the total number of user directory requests fluctuate throughout your business day
- How the user directory requests translate into requests per second over a sustained period.

Complete the following steps to estimate the sustained user directory search rate:

1. Use the authentication guidelines to estimate the number of user directory requests that the authentication load creates.
2. Use the authorization guidelines to estimate the number of user directory requests that the authorization load creates.
3. Estimate the sustained user directory search rate.

Use Authentication Guidelines to Estimate Directory Searches

A Policy Server makes a number of user directory requests to service each authentication request. Some of the user directory requests are required, while others can be avoided.

Estimate the number of Policy Server requests that each authentication creates using the following guidelines:

(Required) Two searches to authenticate each user:

- One search/query, per store, to identify the user

- One search/query to verify the user credentials

(Optional) Additional searches may be required depending on how you design policies:

- One search/query for each policy that is bound to a rule that fires when a user is authenticated (OnAuth rule).
- One search/query for each policy that is bound to a response that returns user attributes.

Use Authorization Guidelines to Estimate Directory Searches

A Policy Server makes a number of user directory requests to authorize a user. Some of the user directory requests are required to determine policy membership, while others are dependent on policy design. You can estimate the number of Policy Server requests that each authorization creates using the following guidelines.

- One search/query for each policy in the application or policy domain.

> **Note:** This guideline only applies to policies whose membership filter results in one or more user directory requests. For more information about the relationship between policy membership and user directory requests, see Policy Membership and Authorization Requests.

- One search/query for each policy that is bound to a response that returns user attributes.

> **Note:** For more information about the relationship between responses and user directory requests, see Responses and Authorization Performance.

Note: The user authorization cache can significantly reduce the number of authorization-related requests to user directories.

Estimate the Sustained User Directory Search Rate

The sustained user directory search rate is based on the total number of operations (authentication load plus authorization load), specifically, when and at what rate these requests occur. The chance that these requests are uniformly spread across your business day is unlikely. Rather, the rate at which these requests occur fluctuates, remaining between the lowest and highest (peak) levels for a sustained period.

Estimating the sustained user directory search rate is the process of identifying:

- A sustained period during which the system is servicing an average number of operations.
- How these requests translate into user directory searches.

When estimating the sustained user directory search rate, we recommend using the daily authentication load and authorization load to identify:

- The rate at which total operations occur throughout the day

Note: We recommend beginning with an evaluation period of 24 hours, broken down into one-hour increments. However, depending on the requirements of your enterprise, you can compare your daily results over a period of weeks or months to gain a better understanding of usage throughout the year.

- The sustained period during which the system is servicing an average number of requests
- The approximate number of requests that occur during the sustained period.

Case Estimate the Sustained User Directory Search Rate

The company has determined that:

- The daily authentication load and authorization load for the application result in approximately 888,000 total operations.
- The total operations result in approximately 2,051,520 user directory requests.
- The system is operating at sustained levels for approximately five hours (9:00 AM - 2:00 PM).
- During sustained levels, approximately 84,000 operations occur, per hour.
- Approximately 420,000 (84,000 * 5) operations, or 48 percent (420,000 / 880,000) of the total operations, occur during these hours.

The company uses the following formula to estimate the sustained user store search rate:

(*total_user_directory_requests* * *percentage_of_requests*) / *number_of_hours* / 3600 = *sustained_user_directory_search_rate*

- **total_user_directory_requests**
  Represents the daily number of requests the Policy Server makes to the user directory to service authentication and authorization requests.
- **percentage_of_requests**
  Represents the percentage of total operations that occur when the system is operating at sustained levels.
- **number_of_hours**
  Represents the number of hours when the system is operating at a sustained rate.
- **sustained_user_directory_search_rate**
  Represents the number of requests, per second, the Policy Server makes to the user directory to maintain the sustained rate of operation.

Result: (2,051,520 * 0.48) / 5 /3600 = 54.7 user directory requests per second.

The Policy Server makes 54.7 requests, per second, to the user directory when servicing authentication and authorization requests during sustained levels of operation.

Estimate the Peak User Directory Search Rate

The peak user directory search rate is based on the total number of operations (authentication load plus authorization load), specifically, when and at what rate the system is operating at peak levels. Estimating the peak user directory search rate is the process of identifying when the system is servicing the highest level of operations and how these requests translate into user directory searches.

When estimating the peak authorization rate, we recommend using the metrics that you gathered when determining the sustained authorization rate to determine:

- The hour the system is servicing the highest number operations.
- The approximate number of operations that occur during this period.

Case Estimate the Peak User Directory Search Rate

A company has determined the application results in a total of 888,000 operations per day. These operations result in approximately 2,051,520 user directory searches. Using metrics gathered during a capacity planning exercise, the company has determined that during the single busiest hour, approximately 278,000 operations, or 31 percent of the total operations, occurred.

The company uses the following formula to estimate the peak user store search rate.

(*total_user_directory_requests* * *percentage_of_requests*) / *number_of_hours* / 3600 = *peak_authentication_request_rate*

- **total_authentication_requests**
  Represents the total number of requests the Policy Server sends to the user store.
- **percentage_of_requests**
  Represents the percentage of operations that occur when the system is operating at peak levels.
- **number_of_hours**
  Represents the number of hours in which the system operates at peak levels.
- **peak_user_directory_request_rate**
  Represents the number of requests, per second, that the Policy Server makes to the user store to maintain the peak authentication rate.

Result: (2,051,520 * 0.31) / 1 / 3600 = 176.6 requests per second.

The Policy Server makes 176.6 requests, per second, to the user directory when servicing authentication and authorization requests during peak levels of operation.

Periodic Maintenance Tasks

The following lists details the tasks you can perform for general CA Single Sign-On maintenance. The CA Services implementation team typically covers the details for these tasks, which are based on the specific environment:

- Apply operating system patches.
  Frequency: monthly or as required
- Apply CA Single Sign-On cumulative patches.
  Frequency: monthly or as required
- Monitor the performance of CA Single Sign-On using the OneView Monitor, CA Wily (or an equivalent tool).
  Frequency: continuous
- Monitor the performance of the backend repositories.
  Frequency: continuous
- Back up the backend repositories using native or CA Single Sign-On tools.
  Frequency: dependent on the requirements of your organization
- Maintain the backend repositories using native tools. Examples of this maintenance include the following items:
  - Indexing
  - Backing up the transaction logs to reduce the consumption of disk space.

  Frequency: dependent on the requirements of your organization
- Remove the tombstones of deleted objects from the policy store by running the XPSSweeper utility.
  Frequency: Every 24 hours. This schedule helps reduce the size of the policy store.
- Archive log files.
  Frequency: dependent on the requirements of your organization
- Audit policies and adjust/optimize as required.
  Frequency: dependent on the requirements of your organization
- Audit authentication and authorization failures. Escalate the events as required.
  Frequency: continuous

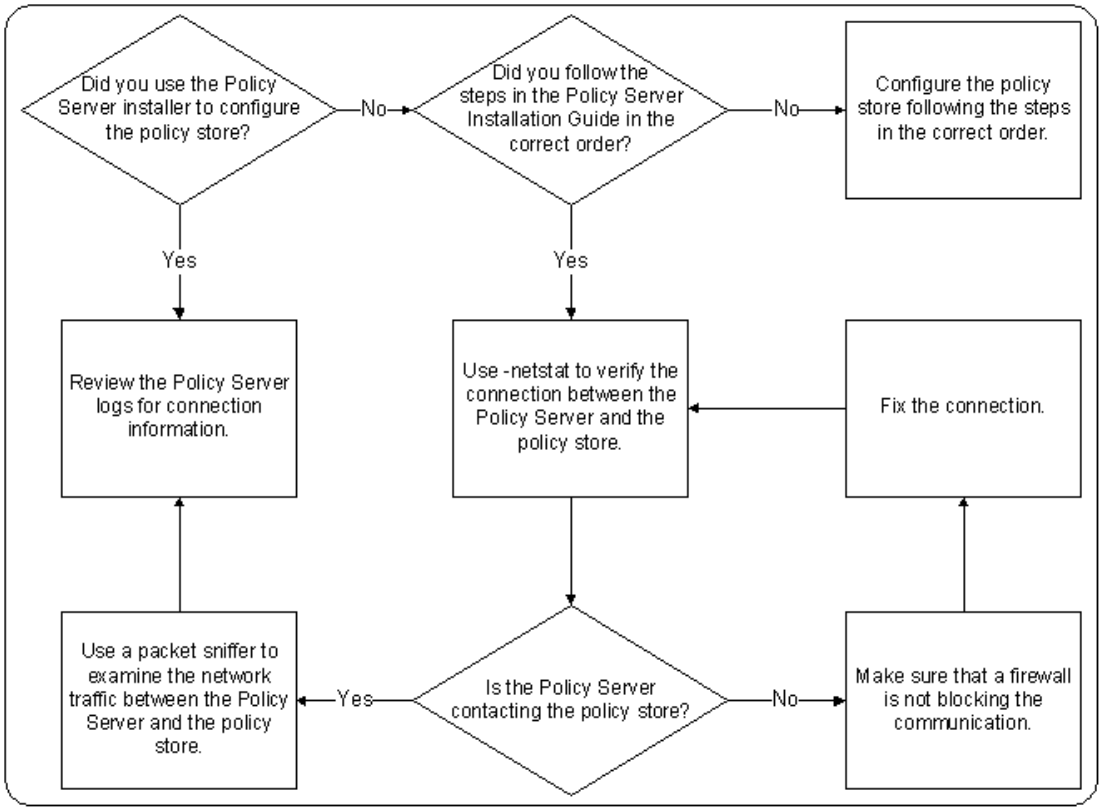
Diagnose Implementation Issues

This section contains the following topics:

Policy Server Policy Store Connection Issues

Various problems are associated with connecting a Policy Server with a properly configured policy store. These problems can range from an incorrectly configured policy store to network and database connections.

Use the following flowchart to diagnose problems:



SM--Policy Server and Policy Store Connection Diagnosis

Consider the following:

- Using a packet sniffer on the Policy Server host system lets you record error messages that the policy store sends to the Policy Server. If a connection request contains an error message stating that the connection was refused, the database or directory server functioning as the policy store is preventing the connection.
- Reviewing the Policy Server logs lets you identify information about the connections the Policy Server is attempting to make. Common reasons the connections fail include:
  - The Policy Server is using invalid administrator credentials to access the policy store.
  - The administrator account that the Policy Server is using does not have read access.

Note: Policy Server logs are located in *siteminder_home*/log.

- **siteminder_home**
  Specifies the Policy Server installation location.

Work with Support

Contents

If you require assistance from the CA Single Sign-On Support team, there is specific information you can gather and include when opening a Support ticket. Including as much information as possible helps to reduce the amount of time it takes the Support team to resolve the issue.

Note: If you are attaching log files as part of your Suppor engagment, be sure that the set of files matches. Also ensure that all the files are from the same time as when the issue occurred.

Environment Information

Gather as much of the following information as possible and include it when you open a Support ticket:

- The operating system on which the Policy Server is installed, including service pack level.
  Example: Windows 2008 SP2

- The web server on which the CA Single Sign-On Agent is installed.
  Example: Windows 2008 SP2, IIS 7.0

- The version, including the service pack and the cumulative release (CR), of the Policy Server.
  Example: r12.0 SP2 CR1

- The version, including the service pack and the CR, of the CA Single Sign-On Agents communicating with the Policy Server.
  Example: r12.0 SP2 CR1

- The policy store type (LDAP/ODBC) and the specific vendor and version.
  Example: Oracle 10g R2

- The specific vendor and version of other CA Single Sign-On data stores.

- If applicable, any other CA products, or third–party products integrated with CA Single Sign-On.
- Any custom code or third–party authentication schemes that are deployed in the environment. Custom code includes code provided by Global Solutions Engineering (GSE) or code developed by your organization.
- Any changes that were recently made to the environment, such as an upgraded CA Single Sign-On component or new hardware.
- When the problem started.

Note: You can use the CA Single Sign-On Platform Support Matrix to verify that the issue is not related to an operating system or third–party product that CA Single Sign-On does not support. For more information, see the CA Single Sign-On Platform Support Matrix.

Log Files

Depending on the problem you are experiencing, Support may request one or more of the following log files:

Component	Files
Policy Server	The Policy Server log (smps.log) The Policy Server profiler log (smtracedefault.log) The audit log (smaccess.log)
Web Agent	The Web Agent log The Web Agent trace log The web server error log The web server access log
WSS Agent	WSS Agent log XML Processing Message Log Web Agent trace log (WSS Agent for Web Servers only) Application server or web server error log Application server or web server access log

Consider the following:

- All Policy Server logs are located in *ps_home*\log.
  - **ps_home**
    Specifies the Policy Server installation path.
- Web and WSS Agent logs have no default location or default names.

Policy Server Crash

If the Policy Server has crashed, the following lists the information that helps Support probe for additional details. This information is not required to open a Support ticket, but is information that Support is likely to request. If you provide this information initially, it can reduce the amount of time it takes Support to resolve the issue.

1. Provide environment information.
2. Describe the problem in as much detail as possible. For example:
   - How often the process is crashing.
   - The number of times the crash has occurred.
   - A description of what was happening on the server when it crashed.
   - The steps to reproduce the crash.
3. Attach the UNIX core file or Windows dump file. If you are attaching these files, consider the following:
   - (UNIX) If possible, provide a packaged core.
   - (Windows) Be sure that this file is a full dump file and not mini dumps produced by a program error debugging tool.
4. Attach the policy store data.
5. Attach the Policy Server log and the Policy Server audit log.
6. Modify the Policy Server trace log output.
7. Attach the Policy Server profiler log.

Attach the Policy Store Data

Support is better able to identify the problem by examining the policy store data. Export the policy store and attach the CA Single Sign-On data information file (smdif) to the ticket.

Modify the Policy Server Trace Log

Support is better able to identify the problem by examining the Policy Server trace log. If the Policy Server is crashing, you can use the Policy Server profiler to capture the problem.

Note: If the Policy Server is hung, it may not be possible to capture the problem. Instead of using the Policy Server profiler, force a core dump.

The Policy Server profiler uses a default configuration file to log Policy Server actions to a trace log. The default settings include information about components and data:

- Components represent logical groups of actions that the Policy Server executes.

- Data represents the actual pieces of data that the Policy Server must trace.

CA Single Sign-On Support uses component and data settings that are not included in the default configuration file to begin the troubleshooting process. Modify the default settings before submitting the Policy Server trace log.

Example: Modified Components

Modify the default trace configuration to include the following components:

- Server
  The Server component includes additional subcomponents. After you add the Server component, remove the following subcomponents:
  - Policy_Object
  - Policy_Object_Cache
  - Administration
  - Audit_Logging
- Tunnel_Service
- JavaAPI

Example: Modified Data Types

Modify the default trace configuration to include the following data types.

Important! The order in which the data types are listed determine the order in which the data is logged. Be sure that the data types are listed in the following order.

- Date
- Time
- Precise Time
- Pid
- Tid
- SrcFile
- Function
- AgentName
- TransactionName

- TransactionID
- Resource
- Realm
- Rule
- Domain
- Group
- Policy
- User
- Directory
- AgentType
- ReturnValue
- ErrorString
- ErrorValue
- AuthStatus
- AuthReason
- AuthScheme
- ClusterID
- RequestIPAddr
- Returns
- Result
- Message

Agent Crash

If an agent has crashed, the following lists the information that helps Support probe for more details. This information is not required to open a Support ticket, but is information that Support is likely to request. If you provide this information initially, it can reduce the amount of time it takes Support to resolve the issue.

1. Gather environment information.
2. Describe the problem in as much detail as possible. For example:

- How often the process is crashing.
- The number of times the crash has occurred.
- A description of what was happening on the server when it crashed.
- The steps to reproduce the crash.

3. Attach the UNIX core file or Windows dump file. If you are attaching these files, consider the following items:
   - (UNIX) If possible, provide a packaged core.
   - (Windows) Be sure that this file is a full dump file and not a mini dump that has been produced by a program error debugging tool.
4. Attach the agent log and the web or application server error log.
5. Attach a tar or zip of the web server binary directory.

   Note: This step does not apply to agents running on an IIS web server.

6. For Web Agents or WSS Agents for Web Servers, attach the Web Agent trace log and the web server access log.

Resource Leaks

If a system resource, such as memory, file handles, network connections, sockets, or disk space, is not being released, the following lists the information that helps Support probe for additional details. This information is not required to open a Support ticket, but is information that Support is likely to request. If you provide this information initially, it can reduce the amount of time it takes Support to resolve the issue.

1. Gather environment information.
2. Describe the problem in as much detail as possible. Include at least the following:
   - The frequency of the resource leak.
   - The size of the resource leak. Measure the resource leak over a time period with a tool that can show the resource allocation, such as prstat.
   - The tool that you used to measure the resource leak.
   - The effect the resource leak has on the system.
     Example: The system crashes or hangs.
   - The steps to reproduce the resource leak or a reproduction test based on the application traffic.

3. Attach logs:
   - (Policy Server) If you are experiencing a Policy Server issue, attach the Policy Server log and the Policy Server audit log.
   - (CA Single Sign-On Agent) If you are experiencing an Agent issue, attach the Agent log and the web server or application server error log.

Functional Issues

A functional issue is defined as an issue where CA Single Sign-On is not performing as detailed by the documentation. If you are experiencing a functional issue, the following lists the information that helps Support probe for additional details. This information is not required to open a Support ticket, but is information that Support is likely to request. If you provide this information initially, it can reduce the amount of time it takes Support to resolve the issue.

1. Gather environment information.
2. Describe the problem in as much detail as possible, including the steps to reproduce the issue.
3. Attach logs:
   - If you are experiencing a Policy Server issue, attach the Policy Server log and the Policy Server audit log.
   - If you are experiencing an Agent issue, attach the Agent log and the corresponding web server or application server error log.
4. Export the policy store to a CA Single Sign-On data information file (smdif) and attach the file.
5. Attach logs:
   - If you are experiencing a Policy Server issue, attach the Policy Server profiler log.
   - (If you are experiencing an Agent issue, attach all Agent logs and the web server or application server access log.

Random Issues

A random issue is defined as an issue that occurs sporadically, and although functional in nature, does not have a pattern that can be reproduced. If you are experiencing a random issue, the following lists the information that helps Support probe for additional details. This information is not required to open a Support ticket, but is information that Support is likely to request. If you provide this information initially, it can reduce the amount of time it takes Support to resolve the issue.

1. Gather environment information.
2. Describe the problem in as much detail as possible. For example:
   - When the issue started.
   - The frequency of the issue.

- The effect the issue has on the system.
  Example: Transactions are taking more time than usual.

3. Attach logs:
   - If you are experiencing a Policy Server issue:
     - attach the Policy Server log with the point of failure, the Policy Server audit log with the point of failure, and the Policy Server profiler log with the point of failure.
     - attach the Policy Server profiler log with the system functioning correctly.
   - If you are experiencing an Agent issue:
     - attach all Agent logs with the points of failure.
     - attach all Agent log with the system functioning correctly.

Locate Knowledge Base Articles

The documentation is only one resource that is available to you. Knowledge base (KB) articles are available on the CA Technical Support site. These articles address various topics related to managing and troubleshooting your environment.

To locate KB articles

1. Log into the Technical Support site (http://www.ca.com/support).
2. Click Support by Product.
   The Support by Product page appears.
3. Locate CA Single Sign-On in the product list and click the link.
   The CA Single Sign-On product page appears.
4. Enter search criteria under Search Support. Search Support is located on the right side of the screen.
   Information matching the search criteria appears.

Measure CA SiteMinder® Performance

Contents

Measuring CA Single Sign-On performance is an iterative process that involves gathering metrics that reflect how different components of your deployment are performing. We recommend measuring round–trip times between each pair of components to determine if performance standards are being met and to identify potential bottlenecks.

Note: Avoid using traditional performance metrics, such as CPU usage, as the sole determining factor for tuning a CA Single Sign-On deployment. For example, the system hosting the Policy Server can be running at low CPU usage under load, but this factor does ensure that the Policy Server has reached optimal performance.

Tools you can use to measure performance include:

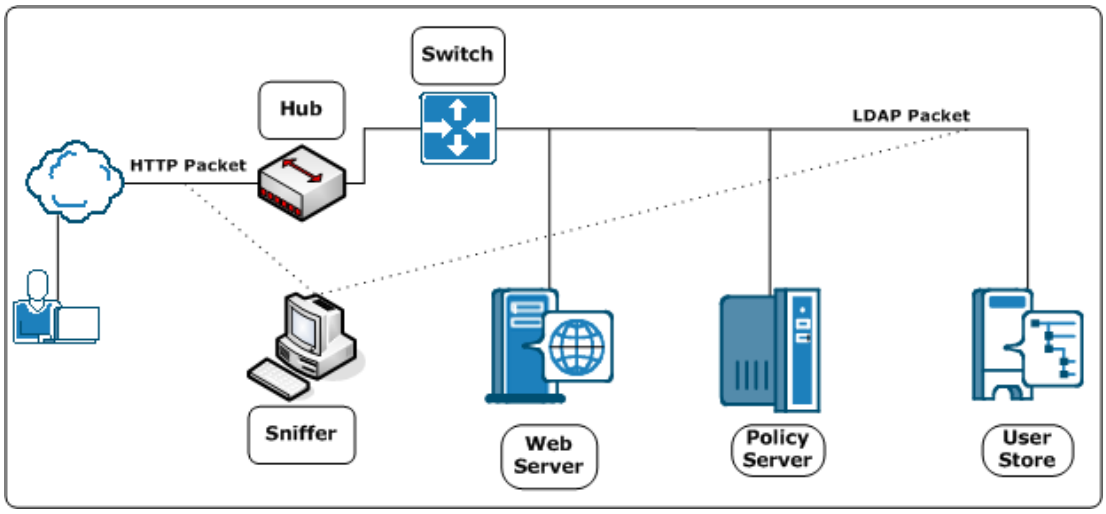
- Network sniffers
- OneView Monitor
- Test Tool
- Directory server utilities and SQL analyzers

Network Sniffers

You can use third-party network sniffers to gather insight into the size and content of a request for unencrypted data without affecting test results. Sniffers can also provide alerts to extra packets being sent, long delays between load balancing, and redirection technologies that logs alone cannot capture.

Note: If the network is set up on a switched hub configuration, place the sniffer between the client and the server on the client–side hub.

The following diagram illustrates a network sniffer in a standard deployment.



SM--Network Sniffer in a Deployment

CA Single Sign-On OneView Monitor

You can use the OneView Monitor to identify performance bottlenecks and gather metrics about resource usage in a deployment. The OneView Monitor also displays alerts when certain events, such as component failure, occur by collecting operational data from the following components:

- Policy Server
- Web Agent

The OneView Monitor can identify performance bottlenecks between Web Agents and the Policy Server by providing metrics such as:

- The average number of authentication attempts and the average time it takes to authenticate a user.
- The average number of authorization attempts and the average time it take to authorize a user.
- The number of cache hits and misses.

CA Single Sign-On Test Tool

You can use the Test Tool to test the interaction between Web Agents and a Policy Server. The Test Tool emulates a Web Agent, which lets you isolate Policy Server performance.

The Test Tool can perform three types of tests:

- **Functionality**
  Tests policies to be sure that they are configured correctly.
- **Regression**
  Tests whether changes, such as migrating a policy store or implementing a new feature, affect the deployment.
- **Stress**
  Test the performance of a Policy Server as it receives multiple requests.

Directory Server Utilities and SQL Analyzers

You can use directory server utilities to simulate Policy Server requests to the directory server or database to isolate query lags. You can also use SQL analyzers to analyze response times between the Policy Server and user directories.

Implementing Federation in Your Enterprise

The content in this section provides CA Single Sign-On Federation deployment models and sample solutions to consider when planning a Federation deployment. Use the Table of Contents to access the content.

CA SiteMinder® Federation Deployments

CA Single Sign-On Federation has two deployment models:

- Partnership Federation
  Partnership federation is based on configuring partnerships between enterprises based on federation standards. The partnership model does not require configuration of CA Single Sign-On-specific objects, such as domains, realms, and policies. This model is recommended for new configurations using CA Single Sign-On Federation.

- Legacy Federation
  Legacy Federation (formerly Federation Security Services).
  Legacy federation is based on configuring CA Single Sign-On objects, such as affiliate domains, authentication schemes, and policies to protect federated resources. This model is primarily for backward compatibility with older deployments.

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 CA Single Sign-On Policy Server, the Administrative UI, and the Web Agent Option Pack to use either model.

Contents

Federation Specifications

CA Single Sign-On 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 OASIS standards (https://www.oasis-open.org/standards) for SAML.

CA Single Sign-On 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

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 WS-Federation specification for communication. For WS specifications and background documentation, and information about ADFS profiles, see the Microsoft documentation.

Entities in a Federated Network

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

Depending on the protocol, these two entities are named differently, but their functions 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 (Partnership)	Identity Provider (IP)	Resource Partner (RP)

WS-Federation (Legacy)	Account Partner (AP)	Resource Partner (RP)

A single site can be the asserting party and the relying party.

Federation Use Cases and Solutions Common to SAML and WS-Federation

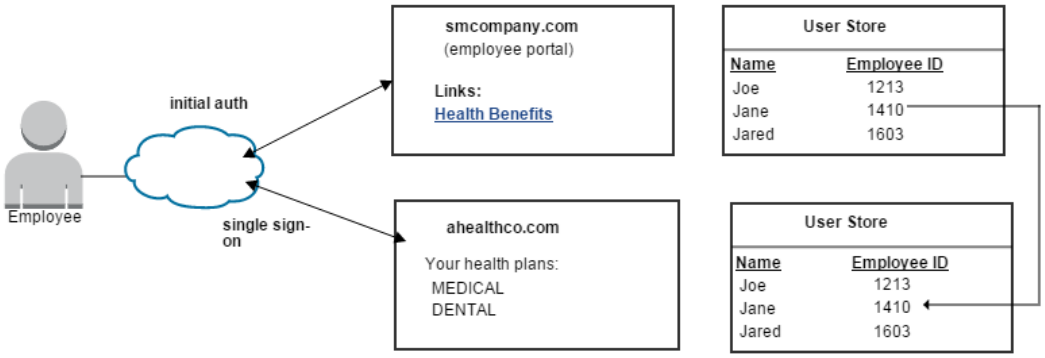
This content provides Federation use cases common to SAML and WS-Federation.

Use Case Single Sign-on Based on Account Linking

In this use case, smcompany.com contracts with a partner, ahealthco.com to manage employee health benefits.

An employee of smcompany.com authenticates at an employee portal at the company website, 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 the correct health benefit information without having to sign on to the website.

The following illustration shows this use case.

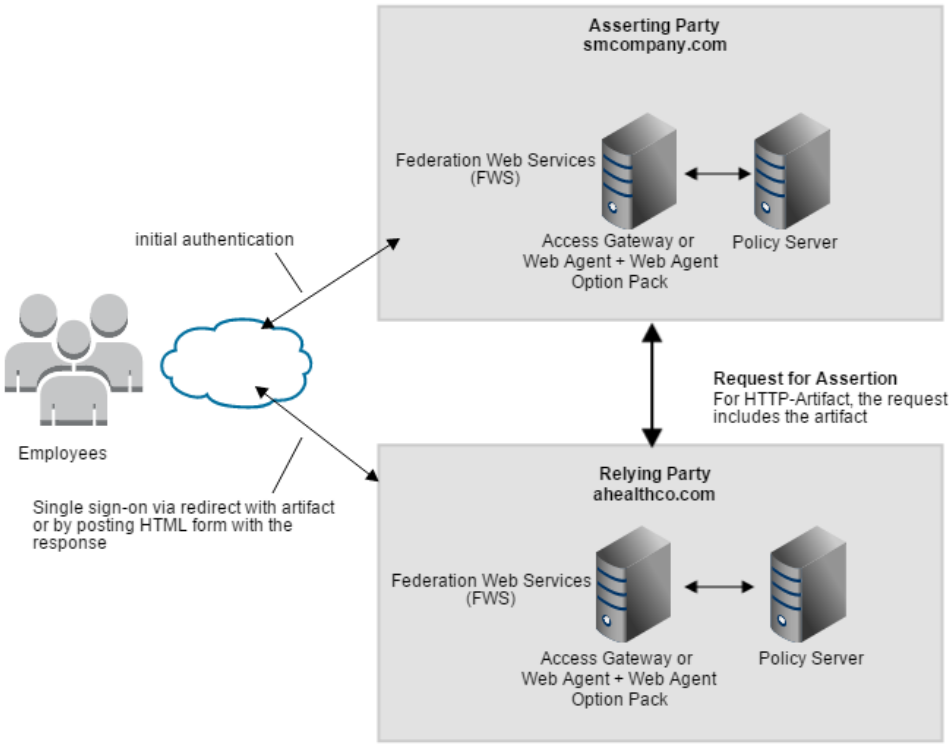


Account linking can be used for browser-based single sign-on, where each partner maintains separate user accounts for the same user. Account linking uses the SAML assertion to associate a federated identifier with the local identity at a partner.

In this use case, ahealthco.com maintains all health-related information and user identities for every employee at 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 Single Sign-on based on Account Linking

Federation can be deployed at smcompany.com and ahealthco.com to solve Use Case: Single Sign-on Based on Account Linking (see page 154).



CA Single Sign-On is deployed at both sites. CA Access Gateway or Web Agent with the Web Agent Option Pack is installed on a webserver system and Policy Server is installed on another system. The installations are the same for smcompany.com and ahealthco.com.

The FWS application provides the servlets which retrieve assertions for the HTTP-Artifact profile and which consume assertions.

Account Linking Solution SAML 1.1 HTTP-Artifact Profile

In this example, smcompany.com is the producer. The administrator at smcompany.com configures a SAML 1.1 producer-to-consumer partnership between smcompany.com and ahealthco.com. This partnership uses the HTTP-Artifact profile for single sign-on.

The partnership at smcompany.com has the following information:

- The location of the assertion consumer service at ahealthco.com.

- The unique Name ID.
- Assertion attributes to be added to the assertion.

An employee of smcompany.com logs into the company site and is initially authenticated by the Web Agent. The employee of smcompany.com accesses the employee portal web site and the sequence of events is as follows:

1. The employee clicks a link at smcompany.com to view health benefits at ahealthco.com. The link makes a request to the Intersite Transfer Service at smcompany.com.
2. The Intersite Transfer Service calls the Policy Server and sends a request that the Policy Server generate an assertion and artifact. The Policy Server generates an assertion and places assertion in the session store. It also generates and returns the artifact to the service.
3. Web Agent or CA Access Gateway redirects the user to ahealthco.com with the SAML artifact.

ahealthco.com is the consumer site. The administrator at ahealthco.com configures a consumer-to-producer partnership with smcompany.com. This partnership uses the HTTP-Artifact profile for single sign-on.

The partnership configuration has the following information:

- The location of the artifact retrieval service at smcompany.com.
- Which attribute in the assertion to use for locating a user in the user directory.
- The search string to locate the user record in the local directory. This record must match the value in the assertion.
- The target resource.

Ahealthco.com receives the assertion and the sequence of events follows:

1. The browser posts the response to the SAML credential collector URL.
2. The service sends a request with the SAML artifact to the assertion retrieval service at smcompany.com. The assertion retrieval service extracts the session ID from the artifact.
3. The assertion retrieval service obtains the assertion from the session store. It sends the assertion as an artifact response to the SAML credential collector at ahealthco.com.
4. The SAML credential collector validates the assertion. The Policy Server creates a session and places a session cookie in the browser for the ahealthco.com domain.
5. The SAML credential collector redirects the user to the target resources at ahealthco.com.

Account Linking Solution SAML 1.x POST Profile

In this example, smcompany.com is the producer. The administrator at smcompany.com configures a producer-to-consumer partnership. The partnership uses SAML 1.x POST profile for single sign-on.

The partnership configuration has the following information:

- The location of the assertion consumer service at ahealthco.com.
- The unique Name ID.
- Assertion attributes to be added to the assertion.

When an employee of smcompany.com accesses the employee portal site, the sequence of events is as follows:

1. Web Agent or CA Access Gateway provides the initial authentication.
2. 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 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 placed in an auto-POST HTML form and sent to the user's browser.
5. The browser posts a form containing the response to the Assertion Consumer Service at ahealthco.com..

ahealthco.com is the consumer site. The SAML credential collector service at ahealthco.com handles the SAML response. The administrator at ahealthco.com configures a consumer-to-producer partnership with smcompany.com that uses the SAMl 1.1 HTTP-POST profile for single sign-on.

The partnership configuration has the following information:

- The location of the artifact retrieval service at smcompany.com.
- Which attribute in the assertion to use for locating a user in the user directory.
- The search string to locate the user record in the local directory. This record must match the value in the assertion.
- The target resource.

The sequence of events is as follows:

1. The SAML credential collector receives the assertion from the producer.
2. The SAML credential collector calls the Policy Server at ahealthco.com.
3. The Policy Server verifies the signature of the assertion then uses it to authenticate the user.
4. After successful authentication, the Policy Server creates an SMSESSION cookie and places it in the browser
5. The browser redirects the user to the target resource at ahealthco.com.

Account Linking Solution SAML 2.0 Artifact Profile

In this example, smcompany.com is the Identity Provider. The administrator at smcompany.com configures an IdP-to-SP partnership with ahealthco.com as the remote SP.

The partnership configuration contains the following information:

- The location of the assertion consumer service at ahealthco.com.
- The unique Name ID.
- The assertion attributes to be added to the assertion.

An employee accesses the employee portal site and the following sequence of events occurs:

1. Web Agent or CA Access Gateway provides the initial authentication.
2. The user clicks a link to view health benefits at ahealthco.com. As the request is initiated at the Identity Provide, the request triggers an unsolicited response.
3. FWS requests the SAML artifact from Policy Server.
4. The Policy Server generates a SAML assertion and an artifact. The Policy Server stores the assertion in the session store, and the artifact as a URL parameter.
5. The Policy Server returns the response containing the SAML artifact to FWS.
6. Web Agent or CA Access Gateway redirects the user with the SAML artifact to ahealthco.com.

ahealthco.com is the Service Provider. The administrator at ahealthco.com configures an SP-to-IdP partnership with smcompany.com, which uses the artifact profile. The partnership configuration has the following information:

- The location of the Single Sign-on Service at smcompany.com.
- Which attribute in the assertion to use for locating a user in the user directory.
- The search string to locate the user record in the local directory. This record must match the value in the assertion.
- The target resource.

The sequence of events is as follows:

1. The Assertion Consumer Service receives the artifact. The service obtains the location of the Artifact Resolution Service at smcompany.com from its partnership configuration.
2. The Assertion Consumer Service makes a call across the back channel to the Artifact Resolution Service at smcompany.com.
3. The Policy Server retrieves the assertion from the session store and returns the response to the Assertion Consumer Service at ahealthco.com.

4. The Assertion Consumer Service validates the response and creates the session for ahealthco.com. A session cookie is written to the browser.

5. The browser redirects the user to the target resource at ahealthco.com.

Account Linking Solution SAML 2.0 POST Profile

In this example, smcompany.com is the Identity Provider. The administrator at smcompany.com configures an IdP-to-SP partnership. The partnership uses SAML 2.0 HTTP-POST profile for single sign-on.

The partnership configuration has the following information:

- The location of the assertion consumer service at ahealthco.com.
- The unique Name ID.
- The assertion attributes to be added to the assertion.

An employee of smcompany.com logs in to the employee portal site.

After a successful initial authentication, the following sequence occurs:

1. Web Agent or CA Access Gateway at smcompany.com initially authenticates the user.
2. The employee clicks a link to ahealthco.com to view health benefits. The Policy Server reads the SAML 2.0 SP configuration.

   The Identity Provider initiates the request, which triggers an unsolicited response.

3. A request is sent to the Single Sign-on (SSO) service at smcompany.com.
4. The SSO service makes a request to policy server to generate a SAML 2.0 assertion or artifact based on the selected profile. For HTTP-POST, the Policy Server generates a SAML assertion.
5. The SSO service receives the assertion response for the selected profile.
6. The signed response is placed in an auto-POST HTML form and sent to the browser.
7. The browser POSTs the response to the Assertion Consumer Service at ahealthco.com.

ahealthco.com is the Service Provider. The administrator at ahealthco.com configures an SP-to-IdP partnership with smcompany.com. The configuration uses the SAML 2.0 HTTP-POST profile for single sign-on.

The partnership configuration has the following information:

- The location of the Artifact Retrieval Service at smcompany.com.
- Which attribute in the assertion to use for locating a user in the user directory.

- The search string to locate the user record in the local directory. This record must match the value in the assertion.
- The target resource.

The sequence of events at ahealthco.com is as follows:

1. The Assertion Consumer Service obtains the response message from the post data.
2. The Assertion Consumer Service reads the IdP configuration to get the target URL.
3. The Assertion Consumer Service passes the signed SAML response as credentials to the Policy Server at ahealthco.com.
4. The Policy Server verifies the signature and then authenticates the user.
5. The login is successful.
6. The Policy Server creates an SMSESSION cookie for the ahealthco.com domain and places the cookie in the browser.
7. The browser redirects the user to the target resource at ahealthco.com.

Account Linking Solution WS-Federation Passive Requestor Profile

In this example, smcompany.com is the Identity Provider. The administrator at smcompany.com configures a WSFED IP-to-RP partnership. The partnership uses the WS-Federation Passive Requester profile for single sign-on. In this use case, ahealthco.com, the Resource Partner, initiates single sign-on.

The SAML token type is SAML 1.1. This part of the IP entity configuration.

The partnership configuration has the following information:

- The location of the Security Token Consumer Service at ahealthco.com.
- The unique Name ID.
- The assertion attributes to be added to the assertion.

When an employee of smcompany.com accesses the employee portal, the following sequence of events occurs:

1. The user visits an unprotected site selection page at ahealthco.com. Web Agent or CA Access Gateway provides the initial authentication.
2. The user clicks a link that points to the Single Sign-on Service at smcompany.com. The browser redirects the user to smcompany.com.
3. The SSO Service calls the Policy Server. The Policy Server generates the assertion.
4. The Policy Server signs the assertion element of the Request Security Token Response and returns a response.

5. The browser POSTS the response in an auto-POST HTML form to the Security Token Consumer Service at ahealthco.com.

ahealthco.com is the Resource Partner.

The partnership configuration has the following information:

- The location of the single sign-on service at smcompany.com.
- Which attribute in the assertion to use for locating a user in the user directory.
- The search string to locate the user record in the local directory. This record must match the value in the assertion.
- The target resource.

The sequence of events is as follows:

1. The Security Token Consumer Service extracts the assertion from the security token consumer response.
2. The service determines the target resource.
3. The Security Token Consumer Service passes the signed assertion as credentials to the Policy Server at ahealthco.com.
4. The Policy Server verifies the signature and then authenticates the user.
5. After successful authentication, the Security Token Consumer Service creates an SMSESSION cookie.
6. The service then places the cookie in the browser and redirects the user to the target resource at ahealthco.com.

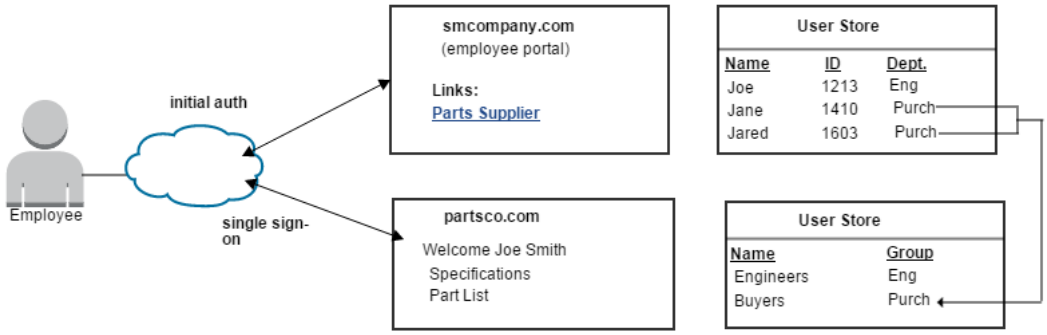
Use Case Single Sign-on Based on User Attributes

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

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

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.

The following graphic shows the relationship between the two partners.



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

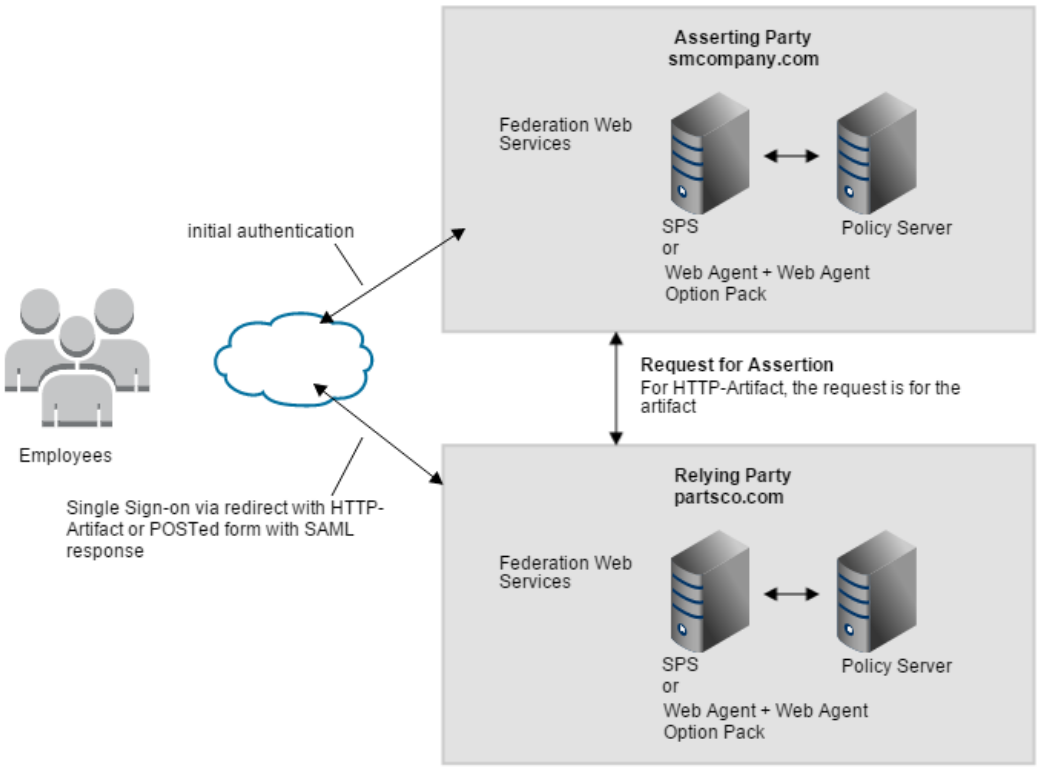
partsco.com does not want to maintain user identities for all smcompany.com employees, but the company wants to control access to sensitive portions of the website. To control the access, partsco.com maintains a limited number of identities for users at smcompany.com. One identity is maintained for Engineers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

Solution Single Sign-on based on User Attributes

Federation can be deployed at smcompany.com and partsco.com to solve Use Case: Single Sign-on Based on User Attribute Profiles (see page 161).

The illustration is similar for SAML 1.1, SAML 2.0, and WS-Federation.



CA Single Sign-On is deployed at both sites. The interaction between the user and each site is similar, where partsco.com is acting as the relying party. FWS application contains all the necessary servlets to process the transaction.

The sequence of events is similar to the solution for Single Sign-on based on Account Linking (see page 155), except for the following items:

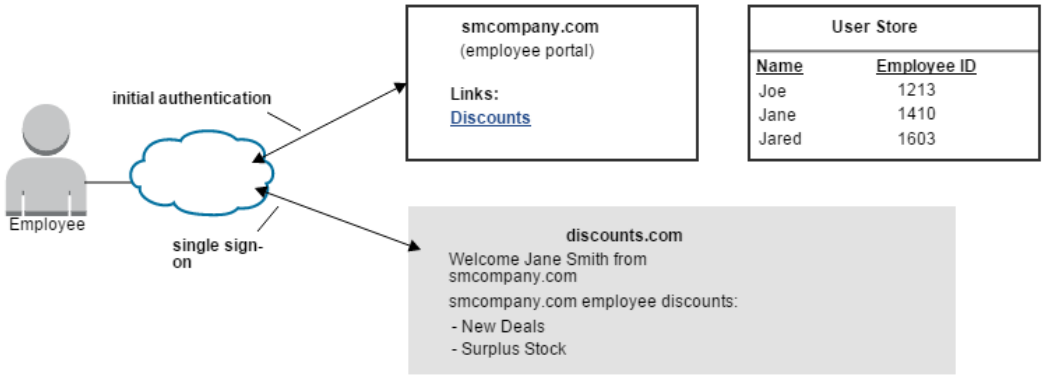
- The administrator at smcompany.com defines the partnership with partsco.com.
- The partnership configuration includes an assertion attribute named *department*. This attribute specifies the department to which the user belongs. The Policy Server includes this attribute in the assertion it generates for the requesting user.
- The administrator defines one user record for each department that is allowed to access the partsco.com website.
- The administrator at partsco.com defines a partnership with smcompany.com.
- The Assertion Consumer Service extracts the department attribute from the assertion. The Policy Server searches the user directory at partsco.com for the user record that has the matching value for the department attribute.

Use Case Single Sign-on with No Local User Account

In this use case, smcompany.com offers employee discounts by establishing a partnership with discounts.com.

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

The following illustration shows this use case.

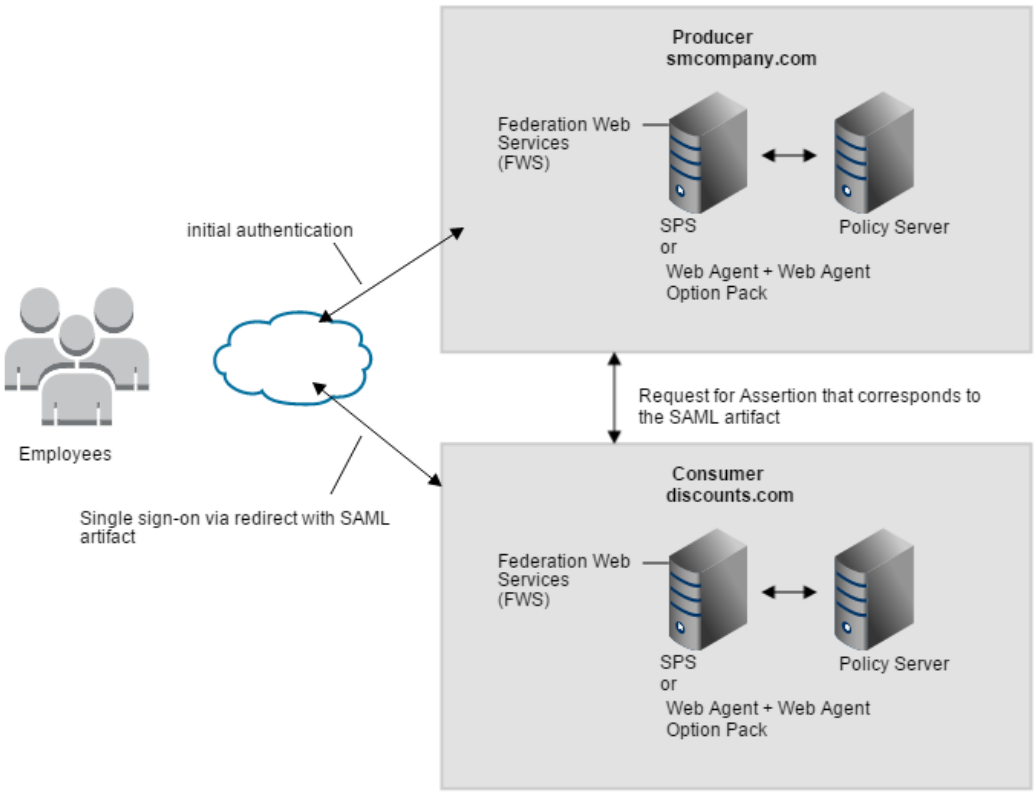


Discounts.com does not maintain any identities for smcompany.com. The company allows all employees of smcompany.com to access discounts.com as long as they have been authenticated at smcompany.com. smcompany.com sends authentication information about the user requesting a resource to discounts.com in a secure manner so access is permitted.

Solution Single Sign-on with no Local User Account

Federation is deployed at smcompany.com and discounts.com to solve Use Case: Single Sign-on with No Local User Account (see page 164).

The following figure shows single sign-on with no local user account. SAML 1.1 is the SSO profile in use.



In this deployment, CA Single Sign-On is at both sites. Smcompany.com is the SAML 1.1 producer. The administrator at smcompany.com configures a SAML 1.1 partnership that includes a remote entity representing discounts.com. Any attributes that are configured in the partnership get included in the assertion.

For this solution to work, every user must map to a single user account, making the single user essentially an anonymous user.

When an smcompany.com employee accesses the employee portal, the following process occurs:

1. Web Agent or CA Access Gateway provides initially authenticates.
2. The employee clicks a link to access deals at discounts.com. This link is referred to as the Intersite Transfer URL because it results in transferring the user to another site.
3. The Intersite Transfer URL makes a request to the Web Agent. This URL contains the location of the SAML credential collector and the target URL at the consumer site.
4. The Web Agent at smcompany.com calls the Policy Server. The Policy Server generates an assertion and an artifact, and stores the assertion in the session store.
5. The Policy Server returns the artifact to the FWS application, which in turn creates a response.
6. The browser redirects the user with the artifact response to discounts.com.

discounts.com is the consumer site. The administrator at discounts.com configures an SP-to-IdP partnership. The partnership configuration specifies the location of the assertion retriever service at smcompany.com, and the protected target resources.

The user identification configuration for the partnership must specify a custom user search specification that looks up a single user. For example, if the user directory is LDAP, the search specification is uid=user1.

Important! To map every user to a single user, a user directory at discounts.com must exist. This user directory must contain a single user record. An alternative is to create a user directory using the Policy Server API that returns the same user record.

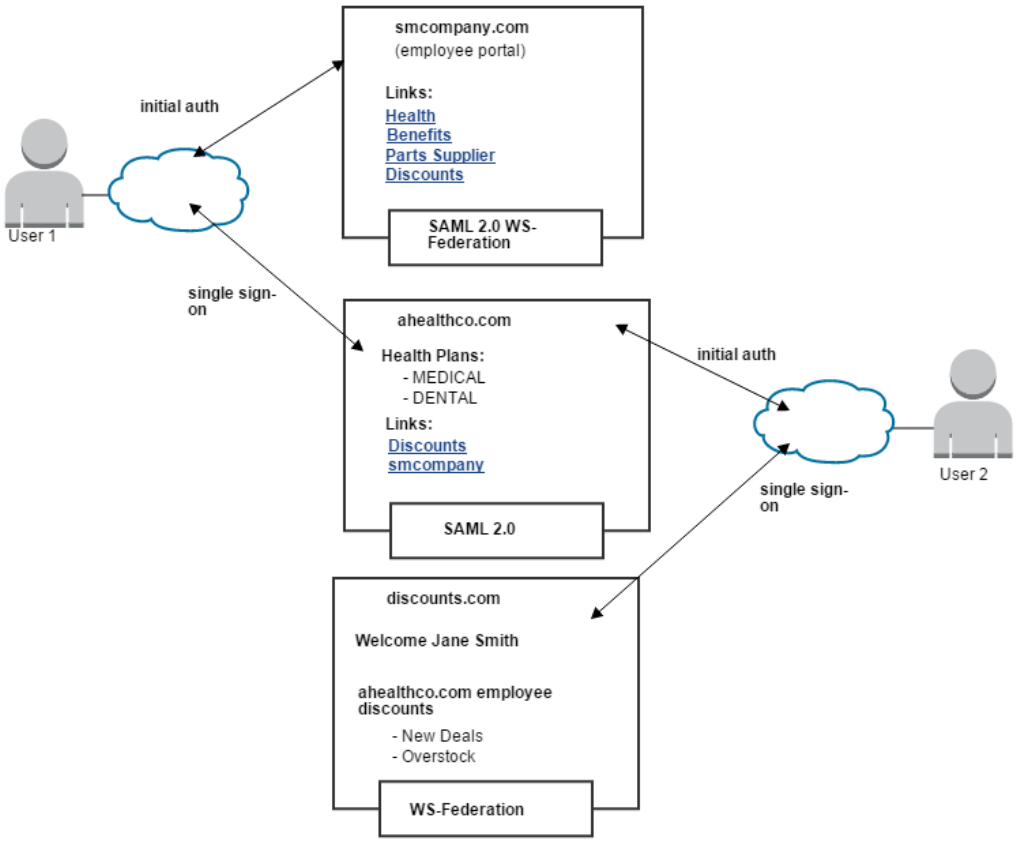
The following process occurs:

1. The browser posts the response to the SAML credential collector, which obtains the location of the assertion retrieval service at smcompany.com.
2. The SAML credential collector makes a back channel call to the Assertion Retrieval Service at smcompany.com. The session ID is extracted from the artifact.
3. The Policy Server retrieves the assertion from the session store and returns it to the SAML credential collector at discounts.com.
4. The SAML credential collector then validates the SAML assertion and issues a session cookie to the browser.
5. The browser redirects the user to the target resource at discounts.com.

Use Case Federation with Multiple SSO Profiles

In this use case, smcompany.com issues assertions for ahealthco.com and discounts.com. Ahealthco.com uses the SAML 2.0 profile. Discounts.com uses the WS-Federation profile. The assertions issued must be generated according to the appropriate profile so that the relying party can consume the assertion.

The following illustration shows the multiprotocol use case.



Solution Federation with Multiple SSO Profiles

The following federation deployment solves the Use Case: Federation with Multiple SSO Profiles (see page 166).

> **Note:** The single sign-on transactions in this solution are similar to the using account linking transactions.

In this solution:

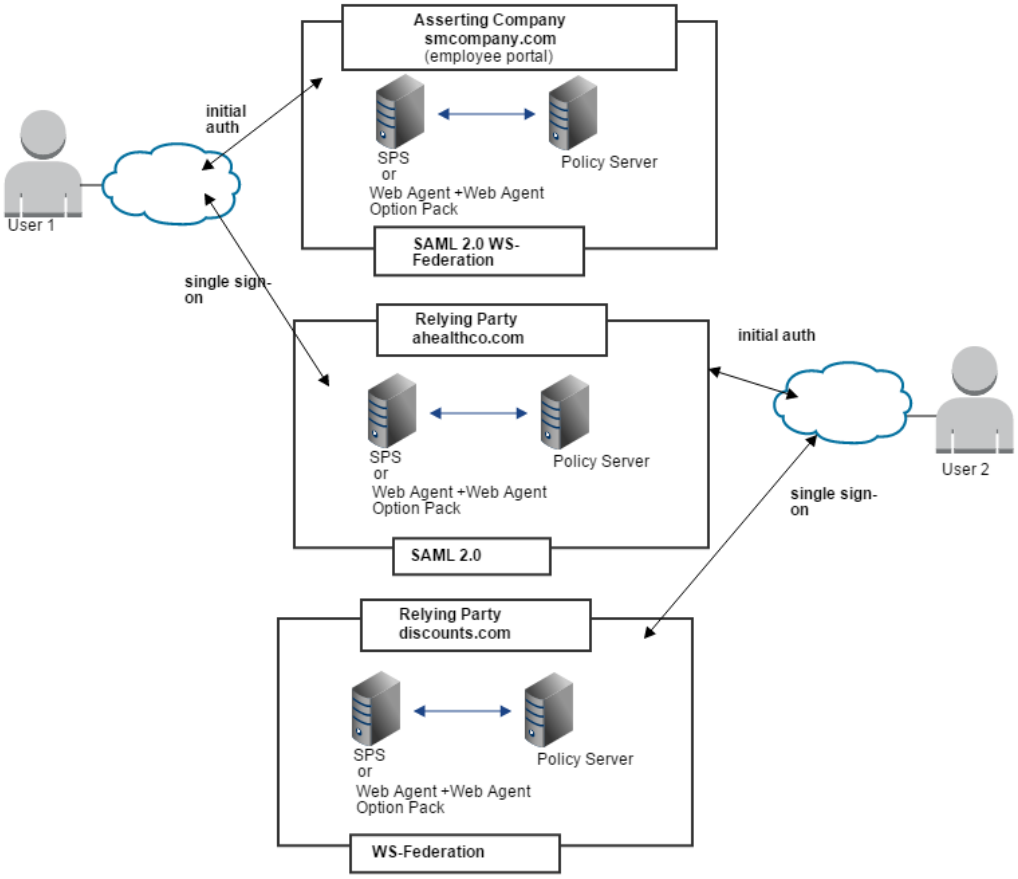
User 1

- smcompany.com is the SAML 2.0 Identity Provider for ahealthco.com.
- ahealthco.com is the SAML 2.0 Service Provider.

User 2

- smcompany.com is the WS-Federation Identity Provider for discounts.com
- discounts.com is the WS-Federation Resource Partner.

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



In this multiprotocol solution, the flow of the single sign-on transactions for the various SSO profiles is similar to the account linking SSO transactions.

- smcompany.com can issue a SAML 2.0 assertion for User 1 to access resources at ahealthco.com.
- Smcompany.com can also issue a token response that includes a SAML 1.1 assertion for User 2 to authenticate at discounts.com. The SSO profile for the assertion is determined by the partnership configuration, and based on the session cookie that is set during initial authentication.

For this solution, the following partnerships are configured at smcompany.com

- An IdP-to-SP partnership, where smcompany.com is the local IdP and ahealthco.com is the remote SP.
- An IP-to-RP partnership, where smcompany.com is the local IP and discounts.com is the remote RP.

The following partnership is configured at ahealthco.com:

- An SP-to-IdP partnership, where ahealthco.com is the local SP and smcompany.com is the remote IdP.

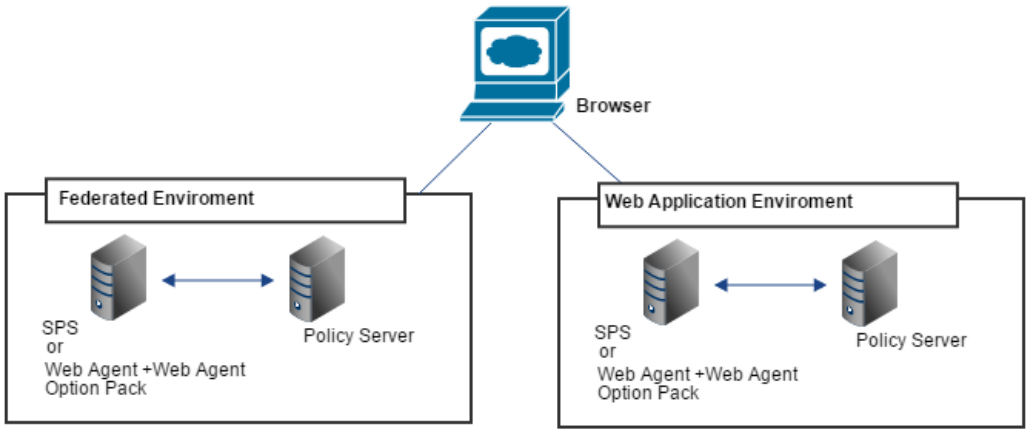
The following partnership is configured at discounts.com:

- An RP-to-IP partnership, where discounts.com is the local RP and smcompany.com is the remote IP.

Use Case SSO Using Security Zones

In this use case, CompanyA protects non-federated web applications and supports federated single sign-on. In a CA Single Sign-On deployment, you cannot have two sessions for a single user that travels from the web application environment to the federated environment. As the user navigates between each environment, the session cookies would overwrite one another.

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

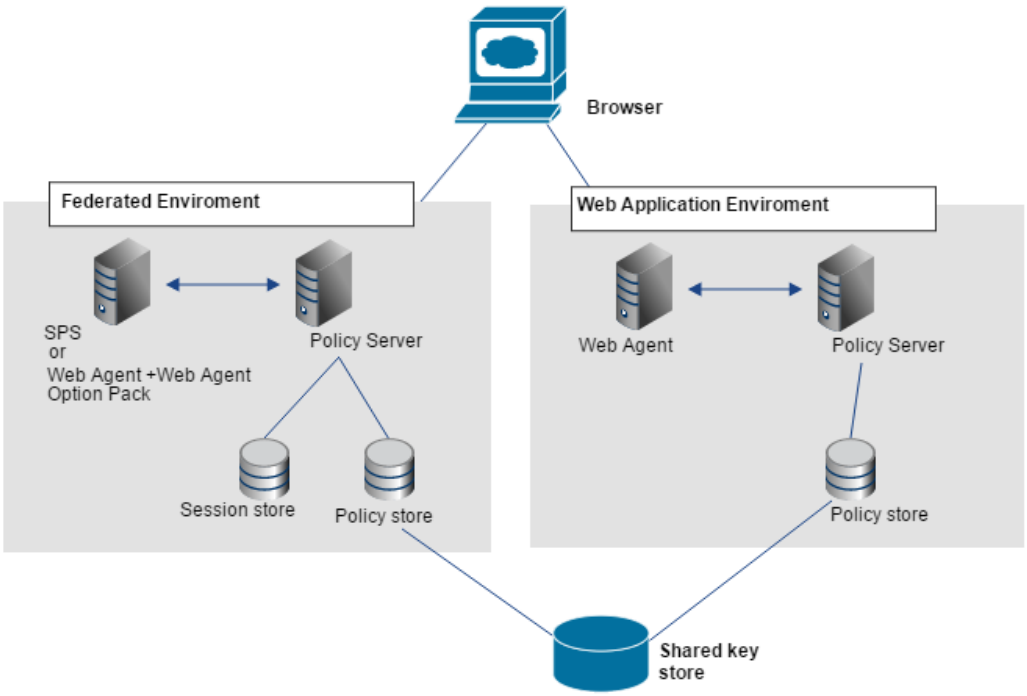


Solution SSO Using Security Zones

This solution illustrates how security zones lets you set up a parallel web application and a federation environment to solve the Use Case: SSO Using Security Zones (see page 169).

A security zone is a segment of a single cookie domain, which is used to partition applications. You can assign different security requirements to each zone. Security zones lets the Policy Server generate different session cookies for a single user in each environment. Though two uniquely named session cookies are generated, each cookie represents the same session across the web application and federated environments. Web agents or CA Access Gateways at the asserting party enforce security zones.

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



The figure reflects the following setup:

Web Application Environment

Agent Configuration Object or Local Configuration file

- DefaultAgent

Trusted Security Zone

- SM (default zone)

Cookies the Web Agent reads for the 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

- FedWA

Trusted Security Zones

- Fed (default zone)
- SM

Cookies the Web Agent reads for the zone

- The FedWA configuration enables Web Agent or CA Access Gateway to read and write SMSESSION cookies.

Note: For this solution to work, each environment must have its own Agent Configuration Object.

The following sequence of events follows:

1. The user logs in to the federation environment.

2. Web Agent or CA Access Gateway in the federated environment directs a request to the Authentication URL to establish a user session.
   The user already has an SMSESSION cookie from a prior authentication in the web application environment.

3. Web Agent or CA Access Gateway in the federated environment reads the SMSESSION cookie. The Policy Server generates a new federation session cookie and Web Agent or CA Access Gateway writes this new session cookie to the browser. The new federation session cookie is based on the SMSESSION cookie.
   Federation requires a persistent session, which is stored in the session store. The SMSESSION cookie that is read from the web application environment is not persistent. When the Policy Server generates a federation cookie, it modifies the cookie and upgrades the session to be a persistent session.

4. The FWS application in the federated environment reads the federation cookie and successfully processes the request for the resource.

SAML 2.0 Federation Use Cases and Solutions

This content provides use cases and solutions for SAML 2.0 Federation.

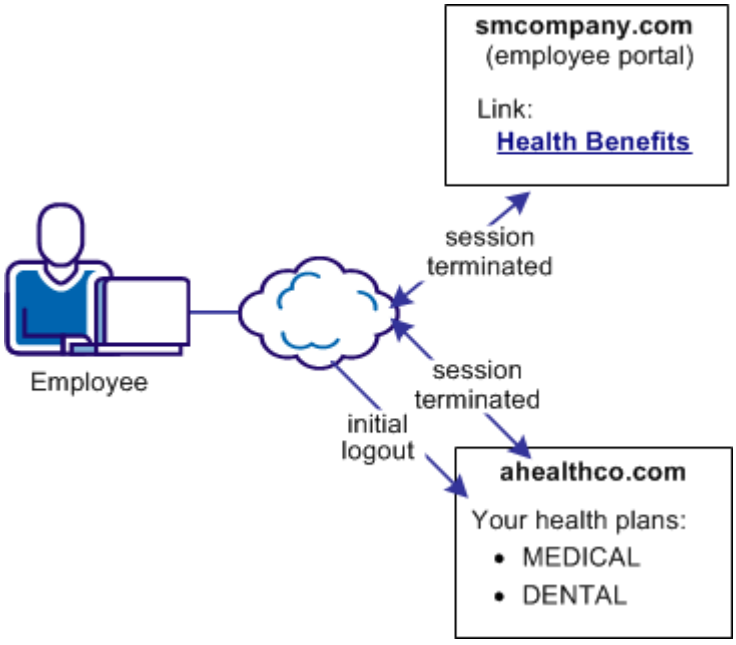
Use Case SAML 2.0 Single Logout

In this use case, an employee of smcompany.com authenticates at the employee portal 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: In this case, the initial logout occurs at ahealthco.com and results in both sessions being terminated.

The following illustration shows the use case.



SM--Use Case 5: Single Logout

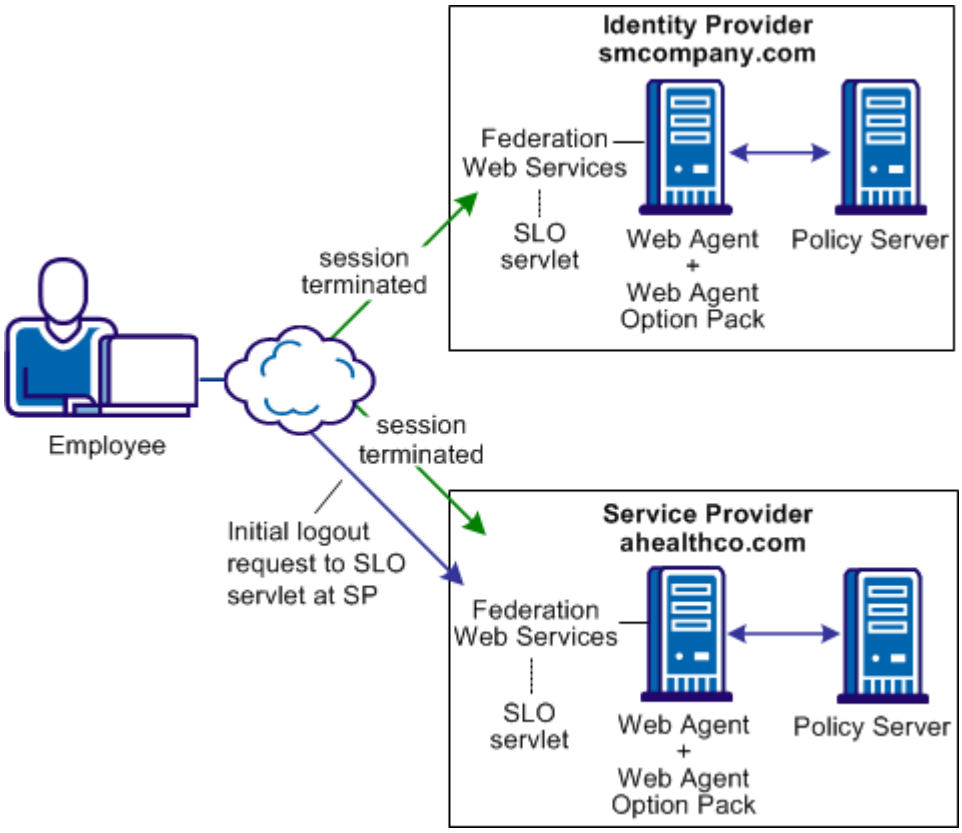
Solution SAML 2.0 Single Logout

You can use federation to solve Use Case: SAML 2.0 Single Logout (see page 171).

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 solution for single logout.



SM--Solution for Single Logout

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

The following sequence of events for SP-initiated single logout occurs:

1. An employee authenticates at smcompany.com then accesses health benefits at ahealthcom.com by way of federated single sign-on. 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 finishes reviewing health benefits and clicks a log-out link at ahealthco.com. The browser accesses the single logout servlet.

3. The FWS application at ahealthco.com renames the existing SMSESSION cookie to SESSIONSIGNOUT to invalidate the current session of the user.

4. The user session is terminated at the ahealthco.com.

> **Note:** The termination does not remove the session from the session store; it sets the state to LogoutInProgress.

5. The Policy Server generates a logout request to invalidate the user session at smcompany.com. The Policy Server also returns the provider ID of smcompany.com.
6. The browser redirects the log out request to the single logout servlet at smcompany.com, with the logout request message added as a query parameter.
7. The FWS application receives the log out request message, and renames the SMSESSION cookie to SESSIONSIGNOUT.
8. FWS invalidates the user session from all Service Providers that are associated with that user session. The only exception is the session at ahealthco.com, who initiated the log out request.
9. After all the Service Providers confirm the logout, smcompany.com removes the user session from its session store. FWS deletes the SESSIONSIGNOUT cookie.

> **Note:** Other Service Providers are not identified in the illustration.

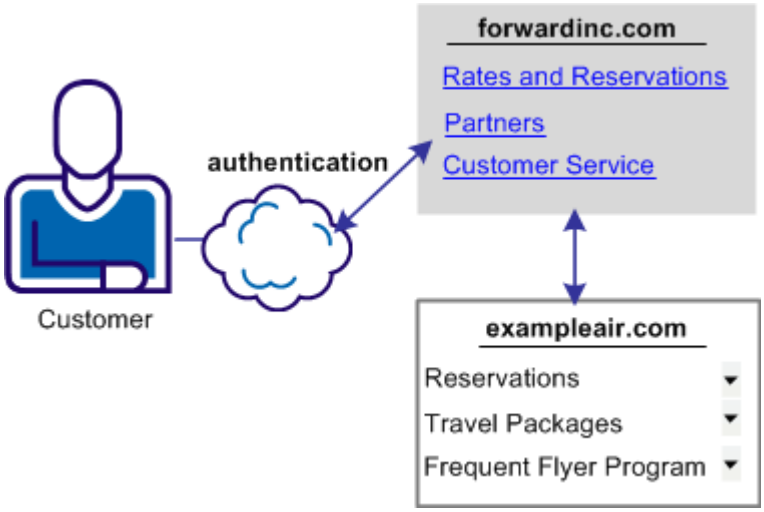
10. Smcompany.com returns a logout response message to ahealthco.com, the initiating Service Provider, and the user session is removed from its session store.
11. The user is finally sent to the SLO configuration page at ahealthco.com.

Use Case SAML 2.0 User Authorization Based on a User Attribute

In this use case, forwardinc.com is a car rental service and exampleair.com is a travel agency.

A customer of forwardinc.com logs in and authenticates at forwardinc.com, then clicks a link at the site to get a quote for a car rental. The customer profile at forwardinc.com includes the customer frequent flyer number for exampleair.com. The frequent flyer account determines a certain status level at forwardinc.com. The status level determines which discount offers the customer receives for car rentals.

The following illustration shows this use case.

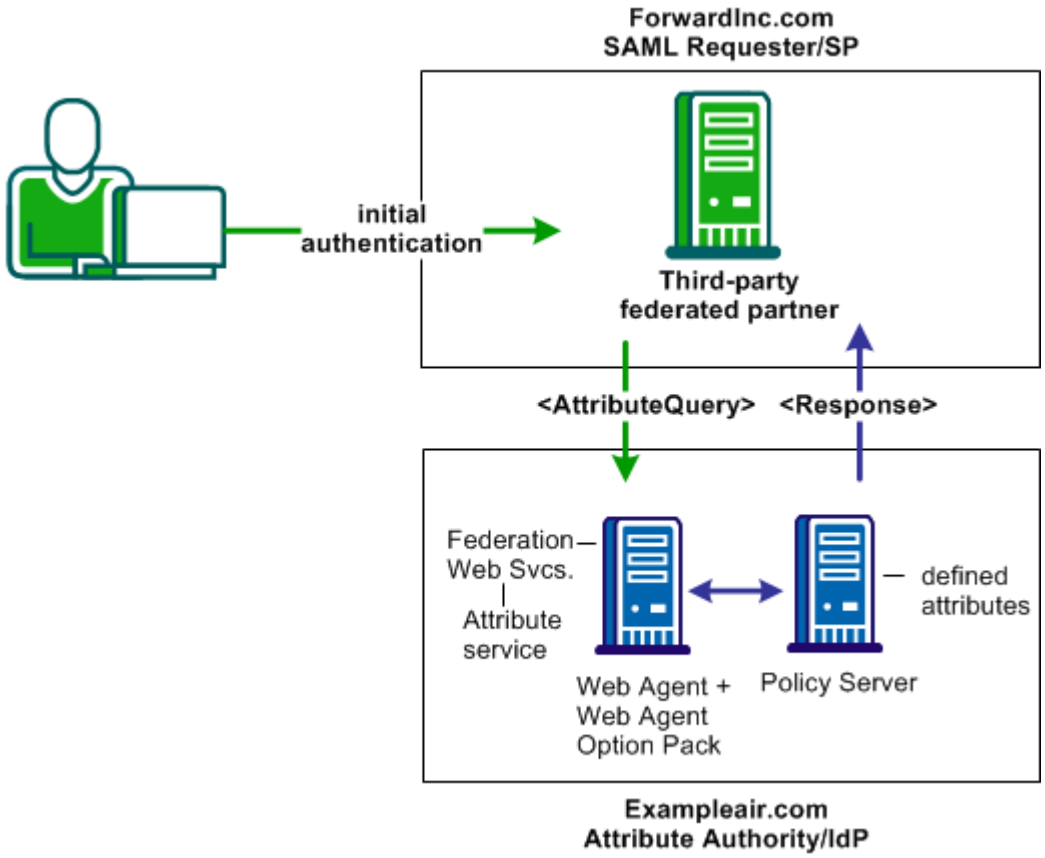


SM--Use Case_User_Attr_Authorization

Forwardinc.com wants to present the appropriate discount information to its customers. However, it does not want customers to log in and authenticate at exampleair.com first then having to log in again at its site.

Solution SAML 2.0 User Authorization Based on a User Attribute

The SAML 2.0 Attribute Query/Response profile can solve Use Case: SAML 2.0 User Authorization Based on a User Attribute (see page 174).



SM--Solution Attribute Authority

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

In this deployment,

- CA Single Sign-On is deployed only at Example.air, the IdP/Attribute Authority. The Web Agent with the Web Agent Option Pack is installed on one system and the Policy Server on another system.

  Note: CA Single Sign-On must be serving as an IdP to implement the attribute query profile. This means that CA Single Sign-On can only be an Attribute Authority and respond to attribute queries. CA Single Sign-On cannot serve as an SP and cannot send attribute queries.

- Forwardinc.com is a third-party Service Provider that is configured to use the Attribute Query /Response profile.

Forwardinc.com is acting as a SAML Requester. When a customer logs in at this site, the following sequence of events occurs:

1. The user logs in to forwardinc.com and authenticates the user.
2. The user clicks a link to rent a car. Forwardinc.com identifies an unresolved frequent flyer attribute.
3. Forwardinc.com tries to resolve the attribute by looking up the user in the local user directory, but it cannot resolve the user attribute variable.
4. Forwardinc.com sends an attribute query as a SOAP request to the IdP/Attribute Authority, exampleair.com. The query request contains the frequent flyer attribute.
5. Exampleair.com looks at its own user directory record for the user and resolves the frequent flyer attribute. Exampleair.com returns an assertion as a SOAP response to forwardinc.com. The assertion contains the requested attribute.
6. The SAML Requester resolves the attribute and authorizes the user for the requested resource.
7. The user is redirected to the target resource.

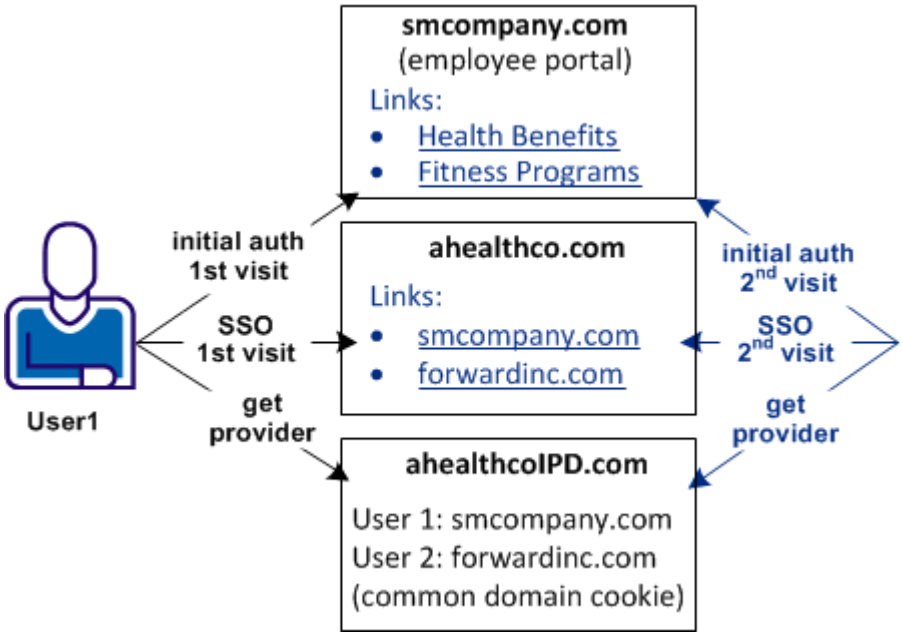
Use Case Identity Provider Discovery Profile

In this use case, several companies contract health benefits from ahealthco.com. A user 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. This profile provides a dynamic way for a Service Provider to determine which Identity Provider has the necessary user record.

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

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



SM--Use Case IdP Discovery (partnership)

In this example, User1 arrives at ahealthco.com. Ahealthco.com determines that User1 came from smcompany.com. Aheathco.com sets a cookie for smcompany.com in the common domain cookie at ahealthco.com. Other companies, such as forwardinc.com is another Identity Provider that uses ahealthco.com as a health provider. When a user from forwardinc.com comes to ahealthco.com, a cookie is set in the common domain cookie also.

Solution Identity Provider Discovery Profile

Federation can solve Use Case: Identity Provider Discovery Profile (see page 177).

The IdP Discovery profile (SAML 2.0 only) is implemented using a cookie domain that is common to the two federated partners. A cookie in the agreed upon domain contains the list of IdPs that a user has visited.

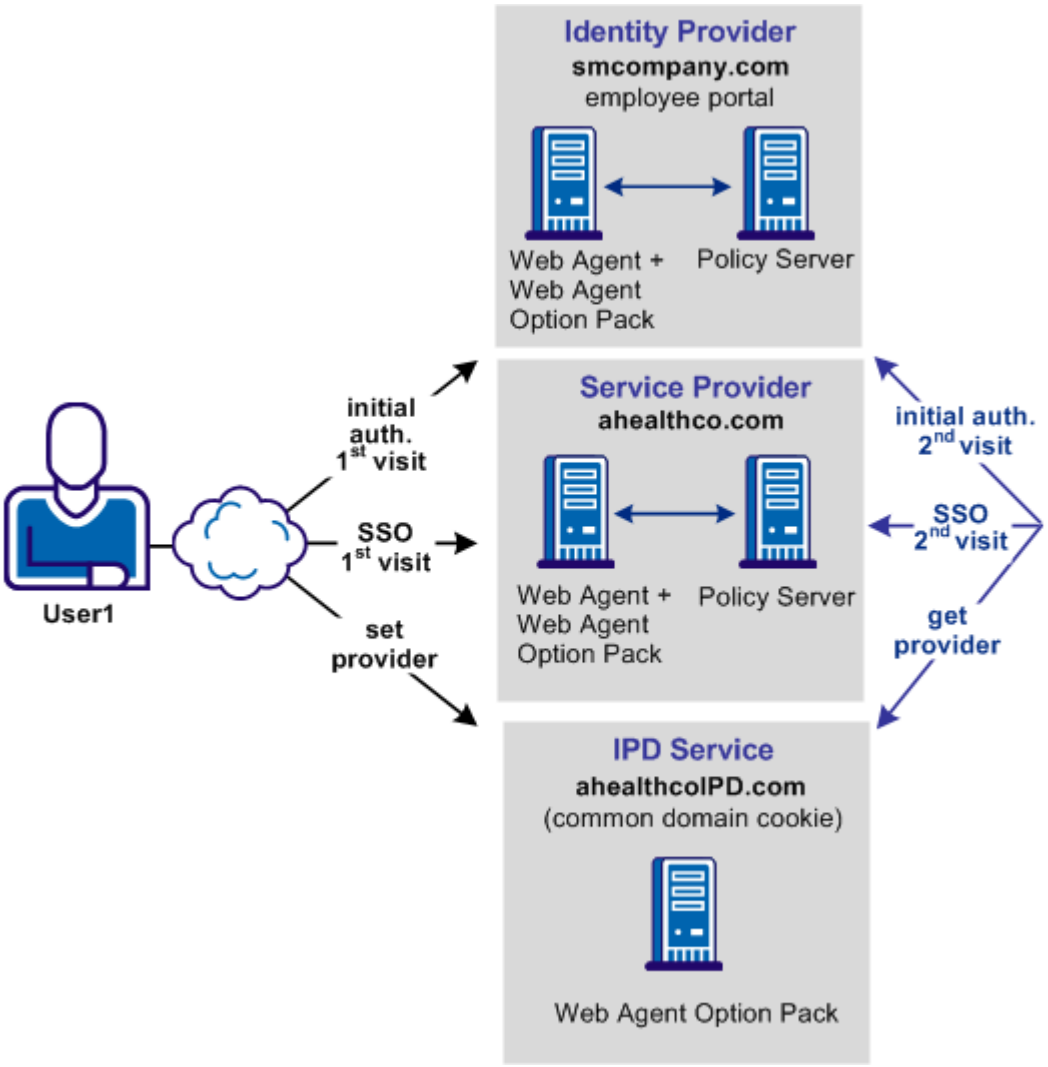
Note: The user that the Service Provider wants to authenticate must have visited the Identity Provider and authenticated before arriving at the Service Provider.

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. This site has SAML 2.0 SP-to-IdP partnerships configured with each Identity Provider that uses its services.

- ahealthcoIPD.com is the Identity Provider Discovery Service for ahealthco.com. The Federation Web Services application installed with the Web Agent Option Pack, provides the IPD service. This service reads the common domain cookie, which includes all relevant Identity Providers for ahealthco.com.

The following illustration shows the federated network for this solution.



SM--Solution for IdP Discovery (partnership)

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

The transaction flow is as follows:

1. User 1 initially logs in and authenticates at smcompany.com. The user then federates to ahealthco.com without having to reauthenticate.
   An agreement exists between smcompany.com and ahealthco.com to use ahealthcoIPD.com as the IPD service.

2. The FWS application at smcompany.com requests the Identity Provider Discovery Profile (IPD) configuration from the Policy Server, passing its Identity Provider ID.

3. The Policy Server returns with the IPD configuration, such as IPD Service URL, common domain cookie, and persistence information of the common domain cookie.

4. The FWS application at smcompany.com redirects the user to the IPD Service URL to set the common domain cookie. The Identity Provider ID of smcompany.com is written to the common domain cookie at the IPD service.

5. The IPD service redirects the user back to the single sign-on service at smcompany.com. The redirect contains an authentication request.

6. The FWS application at smcompany.com requests an assertion from the Policy Server. The Policy Server generates the assertion that is based on the partnership configuration it has for ahealthco.com.

7. The FWS application returns the assertion response back to ahealthco.com.

8. User 1 is now successfully logged on to ahealthco.com and can look at the health benefits. The user finishes reviewing health benefits and logs out.

9. In a separate transaction on a different day, User 1 logs in to ahealthco.com directly. User 1 clicks a link to view healthy benefits again. AhealthIPD.com has cookies for all Identity Providers where User 1 has visited. ahealthco.com calls the IPD discovery service to obtain the Identity Provider IDs.

10. Ahealthco.com presents a site selection page to User 1 to select the company where they can authenticate. User 1 selects smcompany.com.

11. Ahealthco.com sends an authentication request to smcompany.com. The Policy Server at smcompany.com generates an assertion and sends it back to the Assertion Consumer Service at ahealthco.com.

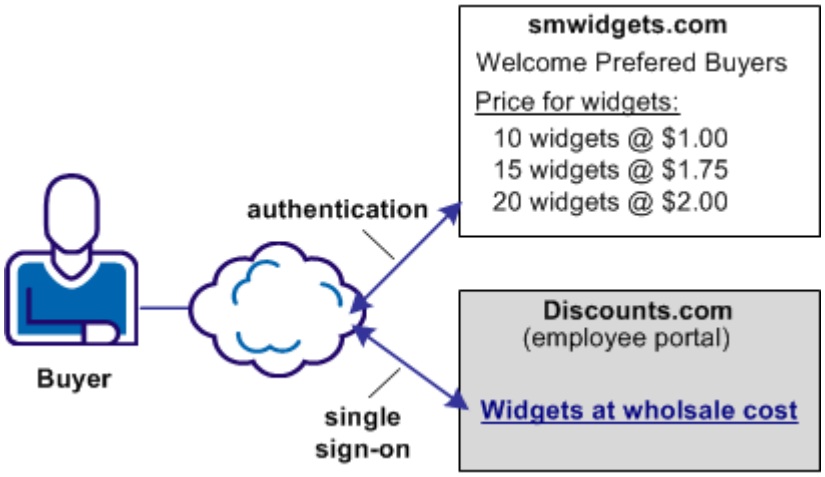
12. The user successfully logs in and is redirected to the requested resource.

Use Case Single Sign-on with No Name ID at the IdP

In this use case, 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.



SM--Use Case No User ID at IdP

There are no buyer identities stored locally at discounts.com, so discounts.com wants to obtain an identity for buyers at smwidgets.com. Discounts.com sends an authentication request to smwidgets.com. Smwidgets.com receives the request, but it cannot find a value for the NameID attribute. It generates a unique persistent identity for the buyer and adds this identity to the assertion. Discounts.com uses this unique identifier to allow the buyer access to the requested resource.

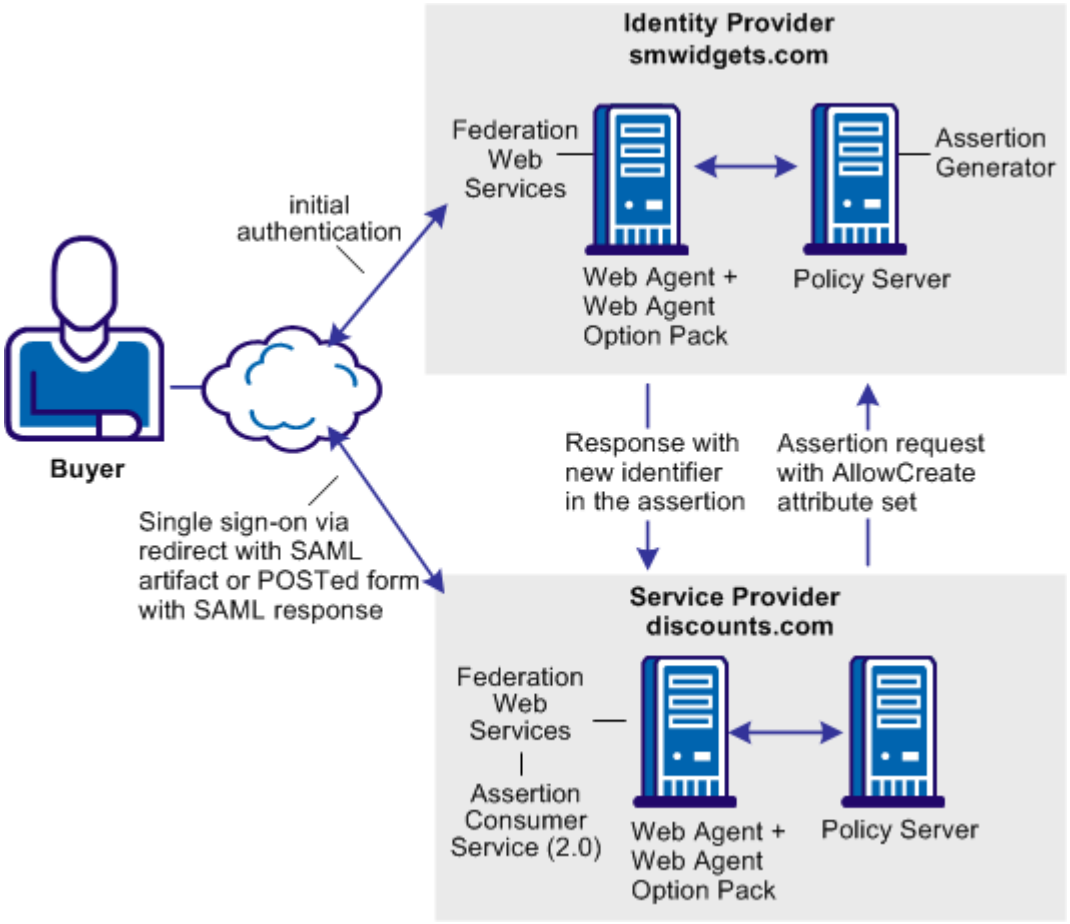
Solution Single Sign-on with No Name ID at the IdP

Use of the Allow/Create attribute solves Use Case: Single Sign-on with No Name ID at the IdP (see page 180).

NOTE: This solution requires the SAML 2.0 profile.

Federation 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 is installed on another system.

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



SM--No User at the IdP

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

To enable single sign-on between the two sites with no user identity at the Identity Provider, the following sequence of events occurs.

1. The user, in this case the buyer, authenticates at discounts.com. This link initiates an authentication request.

2. The Policy Server at discounts.com checks for presence of the Allow/Create option in the configuration. The option is enabled in the SP-to-IdP partnership at the discounts.com.

3. An attribute named AllowCreate is included in the authentication request. The Federation Web Services application at the local Web Agent redirects the authentication request to smwidgets.com.

4. The Policy Server at smwidgets.com generates an assertion. During the assertion generation, the Policy Server searches for the NameID attribute in the user record of the user requesting the resource. There is no value for the NameID in the user record.

5. The Policy Server verifies its configuration for the AllowCreate option. The Policy Server also checks the authentication request for the AllowCreate attribute, which it finds.

6. The Policy Server generates a unique persistent identifier because of the presence of the AllowCreate attribute in the authentication request and in its own configuration. The Policy Server places the identifier in its user store.

7. The Policy Server returns the assertion to the Federation Web Services at the application at the discounts.com. The Federation Web Services application sends an auto-post form containing the assertion response to the Assertion Consumer Service at discounts.com.

8. The Service Provider at discounts.com uses the response message to log in to the Policy Server, using the response as credentials.

9. The Policy Server validates the response by looking for the NameID in its user store. The Policy Server locates the user and logs in the user.

10. The Web Agent generates an SMSESSION cookie for the discounts.com domain.

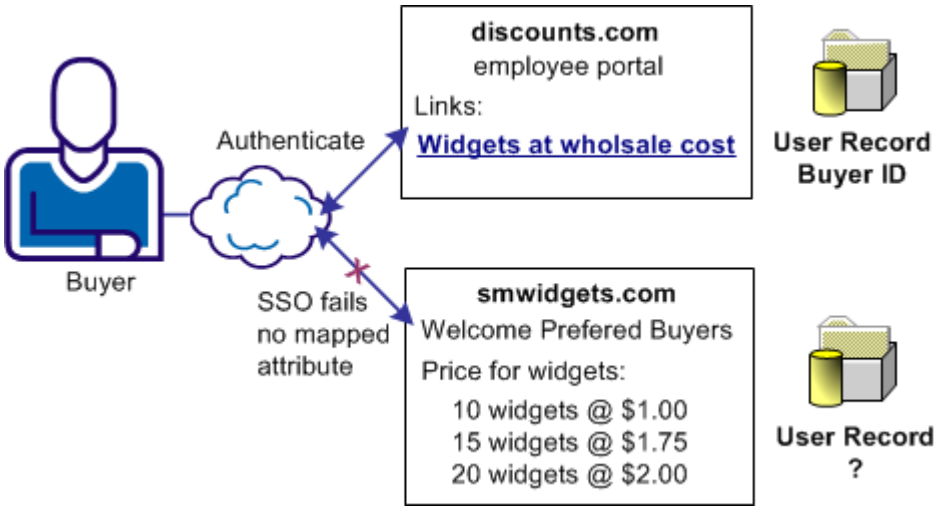
11. The Web Agent places the cookie in the browser and redirects the user to the target destination.

Use Case SSO with Dynamic Account Linking at the SP

In the use case, the IdP, discounts.com, includes an attribute named buyer ID that identifies a particular user. 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. Smwidgets.com must create an attribute in the appropriate user record so that the buyer can authenticate and gain access to the protected resource.

The administrator at the smwidgets.com establishes a mapping using dynamic account linking. The mapping lets smwidgets can authenticate the buyer and can allow access to resources. When a buyer at discounts.com selects a link to access the latest price list on widgets at smwidgets.com, the buyer is logged in without having to reauthenticate.

The following illustration shows this use case.



SM--Use Case Dynamic Liinking at SP

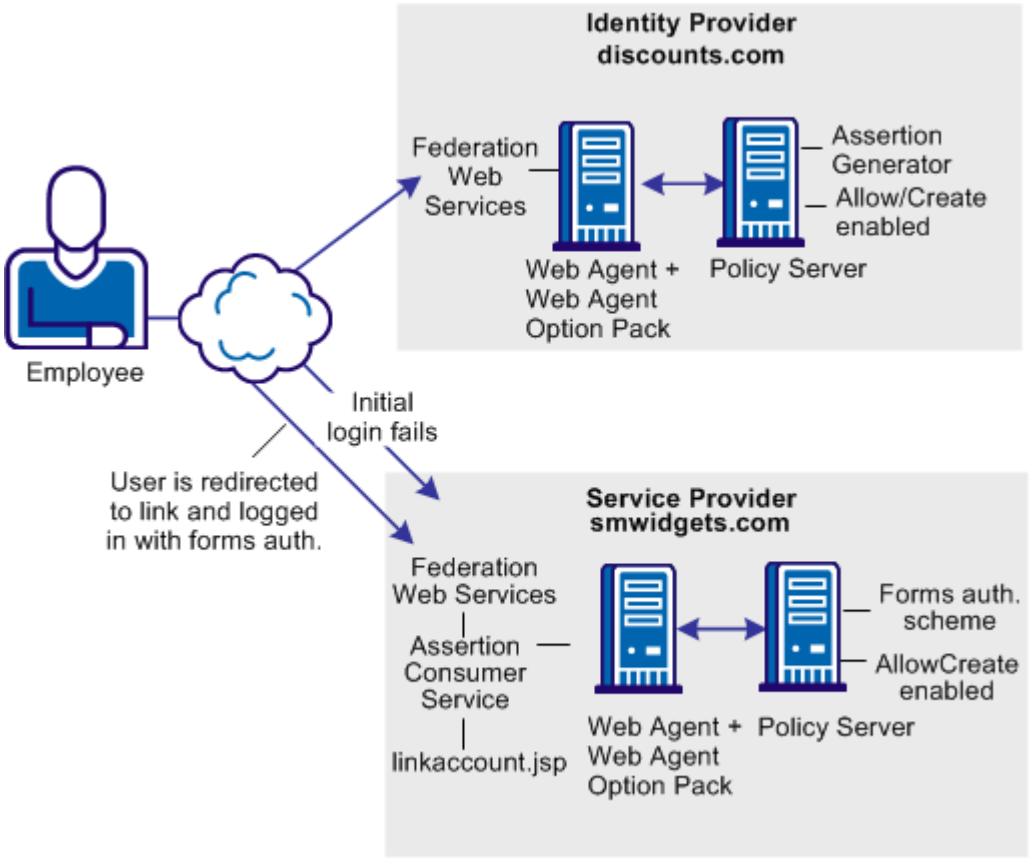
Solution SSO with Dynamic Account Linking at the SP

Federation can be deployed at IdPA.com and SPB.com to solve Use Case: SSO with Dynamic Account Linking at the SP (see page 183).

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

CA Single Sign-On 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.



SM--Solution for Dynamic Account Linking at SP

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

The following sequence of events occurs:

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, but the buyerID attribute does not map to a local user record. The authentication fails.

3. As part of the partnership configuration at smwidgets.com, 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 session at smwidgets.com is established and an SMSESSION cookie is placed in the employee's browser.

5. The linkacount.jsp gets loaded in the browser and the user sees a message to link to the Service Provider 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 partnership indicates which attribute is mapped to the NameID. In this case, the search specification is buyerID=%s.

8. After the attribute is mapped, the user is authenticated based on the assertion. A new user session is established.
   The next time that the same user presents an assertion with the buyer ID, the user successfully gains access to the requested resource.

Configure Dynamic Account Linking at the SP

Configure the following components at the Service Provider to enable dynamic account linking.

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

- 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 scheme protects the redirect URL. The scheme requests the user to log in to create a 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 SAML 2.0 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 CA Single Sign-On 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..

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

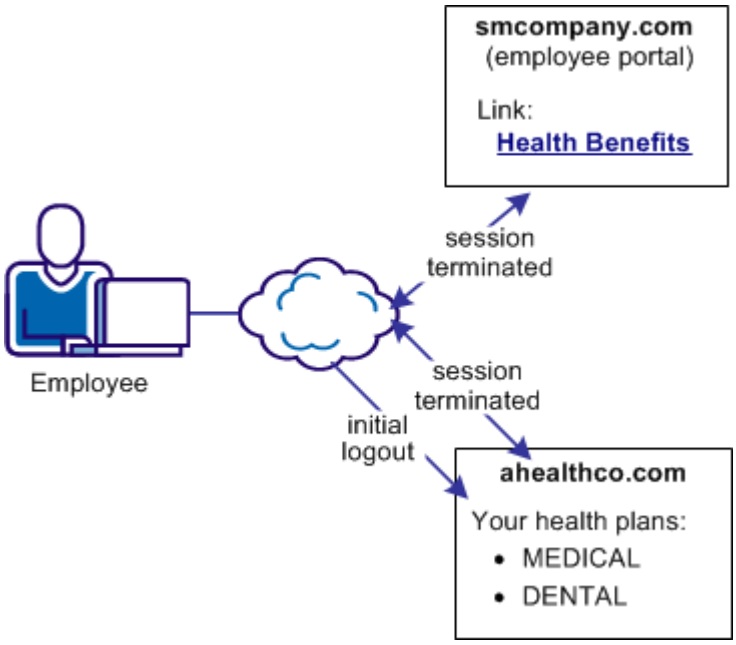
5. Configure a search specification for the SAML authentication scheme. For example, if the Name ID from the assertion replaces buyerID, the search specification would be buyerID=%s.

WS-Federation Signout Use Case and Solution

In this use case, an smcompany.com employee authenticates at the employee portal. The employee 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, ahealthco.com wants the user session at its site and at smcompany.com terminated. Terminating both sessions prevents an unauthorized person from using the existing sessions to access resources at smcompany.com or ahealthco.com.

The following illustration shows the use case.



usecase_single_logout

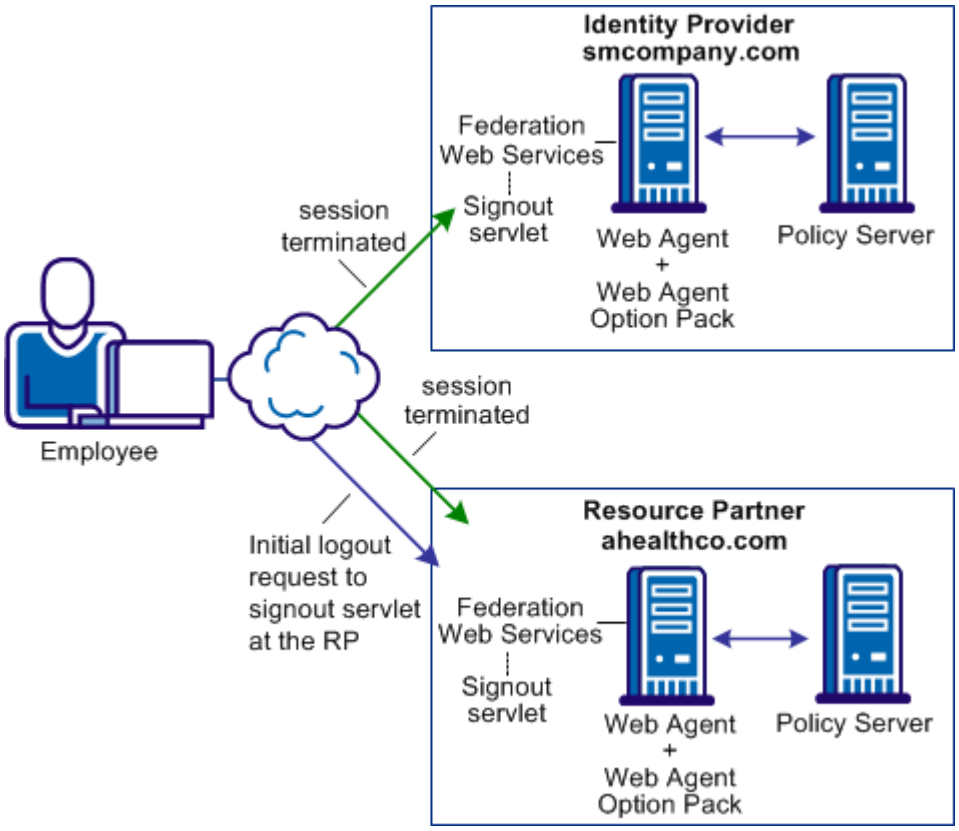
Solution WS-Federation Signout

The following federated deployment solves the Use Case: WS-Federation Sign-out (see page).

In this solution:

- smcompany.com is the Identity Provider.
- ahealthco.com is the Resource Partner, and it initiates the sign-out request.
- A WSFED IP-to-RP partnership is configured between smcompany.com and ahealthco.com to enable single sign-on.
- WS-Federation sign-out using the HTTP binding is enabled at the Identity Provider and the Resource Partner.

The following figure illustrates WS-Federation sign-out.



SM--WSFed Signout Solution

Note: The CA Access Gateway can replace the Web Agent and Web Agent Option Pack to provide the Federation Web Services application functions.

The following sequence of events occurs:

1. An employee authenticates at smcompany.com and federates to ahealthco.com to access their health benefits. 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 finishes viewing his health benefit information, and clicks a log-out link at ahealthco.com. The sign-out service receives a sign-out request.

3. The sign-out service fetches session information from the SMSESSION cookie and determines the Identity Provider that is associated with the user session.

4. The sign-out service calls the Policy Server to invalidate the session.

5. The sign-out service generates a sign-out request, and forwards the sign-out request to the Signout URL at smcompany.com.

6. The sign-out service at smcompany.com receives the request.

7. The sign-out service fetches session information from the SMSESSION cookie and determines the Resource Partner that is associated with the user session.

8. The sign-out service calls the Policy Server to invalidate the session.

9. The sign-out service generates a sign-out request, and posts a sign-out message and multiple RP-SignoutCleanup locations as post data to the SignoutConfirmURL JSP.
   The SignoutConfirm page generates a frame-based HTML page. Each frame contains a Signout Cleanup URL for each Resource Partner that is associated with the user session.

10. Ahealthco.com forwards the sign-out request to any Resource Partner associated with the user session. At each RP, the session is terminated from the session store.

11. Each RP redirects the browser to the Sign-out Cleanup URL ahealthco.com, as the initiating partner to complete the sign-out.

Federation Deployment Considerations

This content provides information to consider when implementing a Federation deployment.

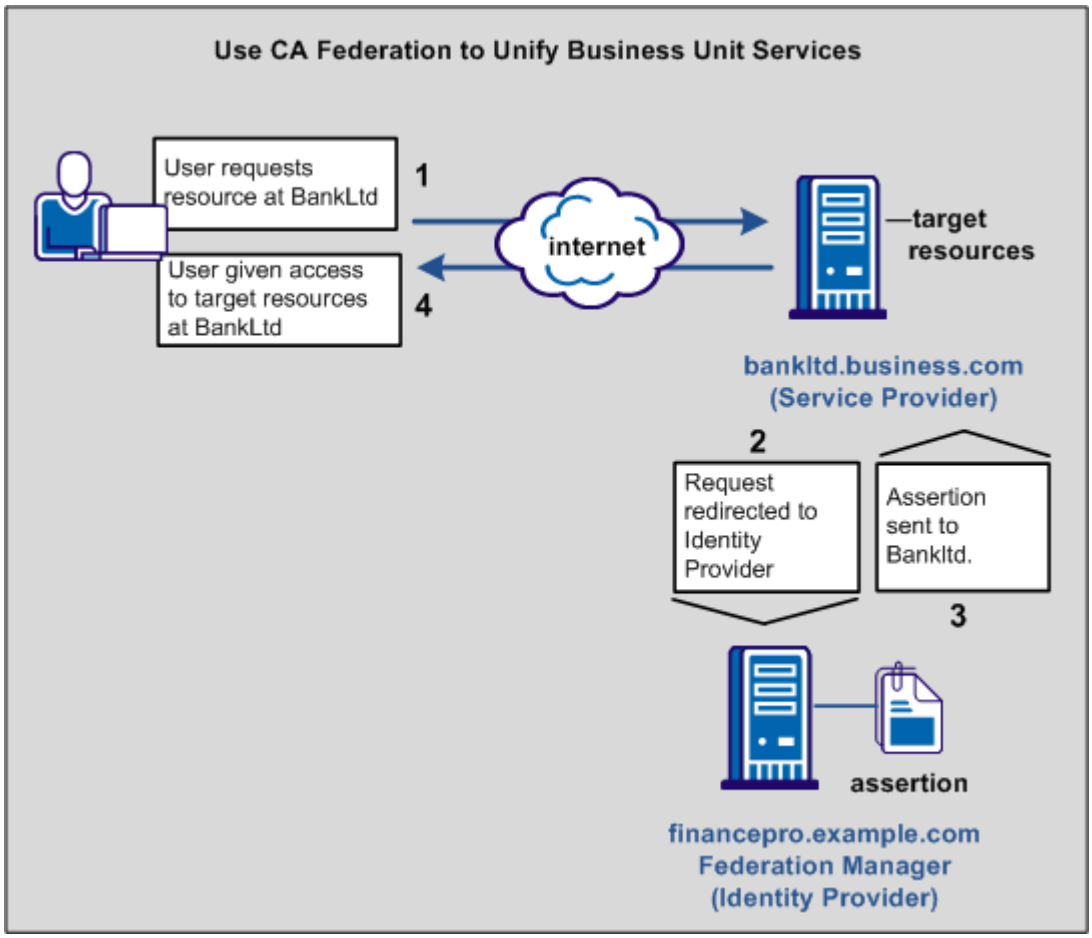
Sample Business Case

A sample business case best illustrates how CA Single Sign-On 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.



FM--sso sample network

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.

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 POST or Artifact, 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.

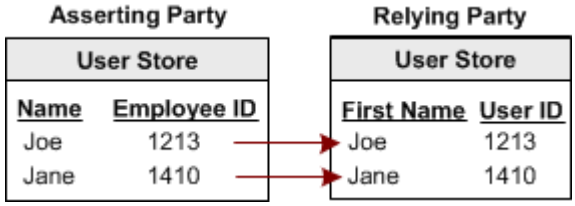
The user identification decision determines what information is sent as the user identity in the assertion.

User Mapping

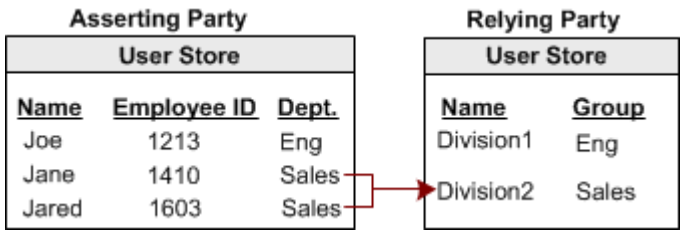
User mapping is the ability to establish a relationship between a user identity at one business and a user identity at another business. Map remote users at the asserting party to local users at the relying party.

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.



- 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:



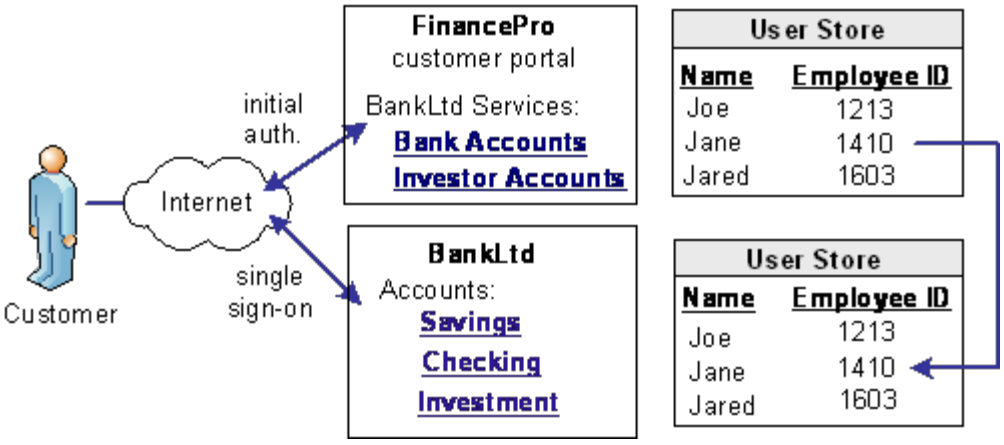
For legacy federation, user mapping is configured as part of a federation authentication scheme. For partnership federation, user mapping is configured as part of the name ID and attribute settings.

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.



CA Single Sign-On Federation 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 CA Single Sign-On Federation 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

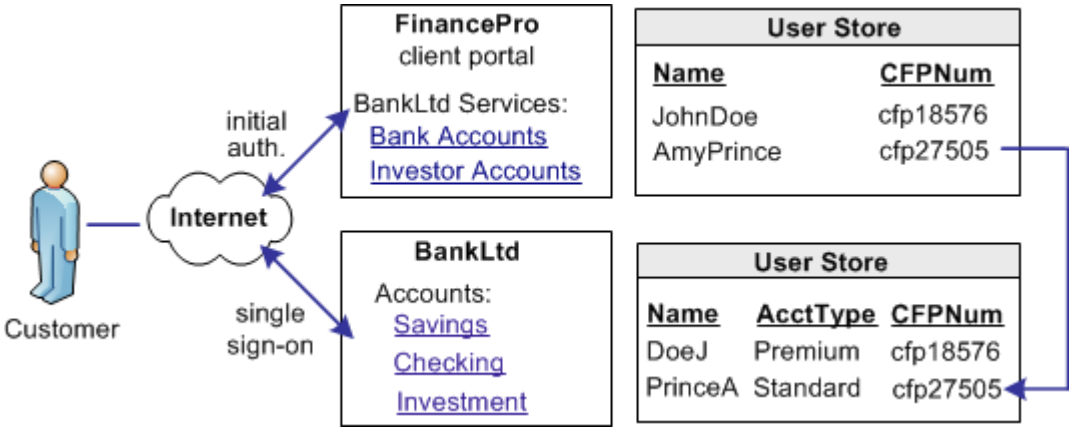
An investor at Financepro authenticates and selects a link to access information at BankLtd. The investor is taken directly to the accounts area of the BankLtd website without having to sign on.

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.



CA Single Sign-On Federation 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.

CA Single Sign-On Federation 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.

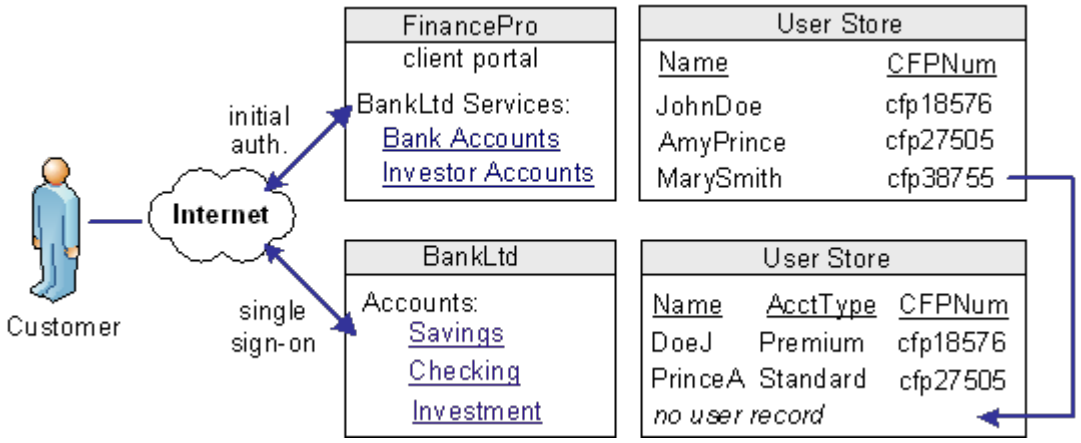
User Provisioning to Establish a Federated Identity (partnership federation only)

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 to implement provisioning to establish the new federated identity for Mary Smith. CA Single Sign-On redirects Mary Smith to the provisioning server at BankLtd. The provisioning application, using the federated identity information, creates a user account in the user store.

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



Federation 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

CA Single Sign-On Federation 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 whether to use SAML or WS-Federation for a partnership depends on the binding that each side supports.

For a new federation, there are no legacy requirements for either partner. Therefore, the recommended SAML 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 deployments use Active Directory Federation Services (ADFS), configure WS-Federation.

Federating with Each CA Single Sign-On Federation 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 in the partnership wizard 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
   Identifies the users for which the asserting party generates assertions/tokens 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 the 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 SLO or Sign-out
   Defines the Single Sign-on 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), authentication context, Enhanced Client or Proxy (ECP) profile, and Identity Provider Discovery profile. For WS-Federation, you can configure sign-out.

5. AuthnContext (SAML 2.0 only)
   Enables the Service Provider to obtain information about the authentication process to establish a level of confidence. This feature also enables the Identity Provider to include the authentication context in an assertion.

6. 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
   - WS-Federation sign-out responses.

7. 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 CA Single Sign-On 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 CA Single Sign-On is the relying party, the configuration steps include:

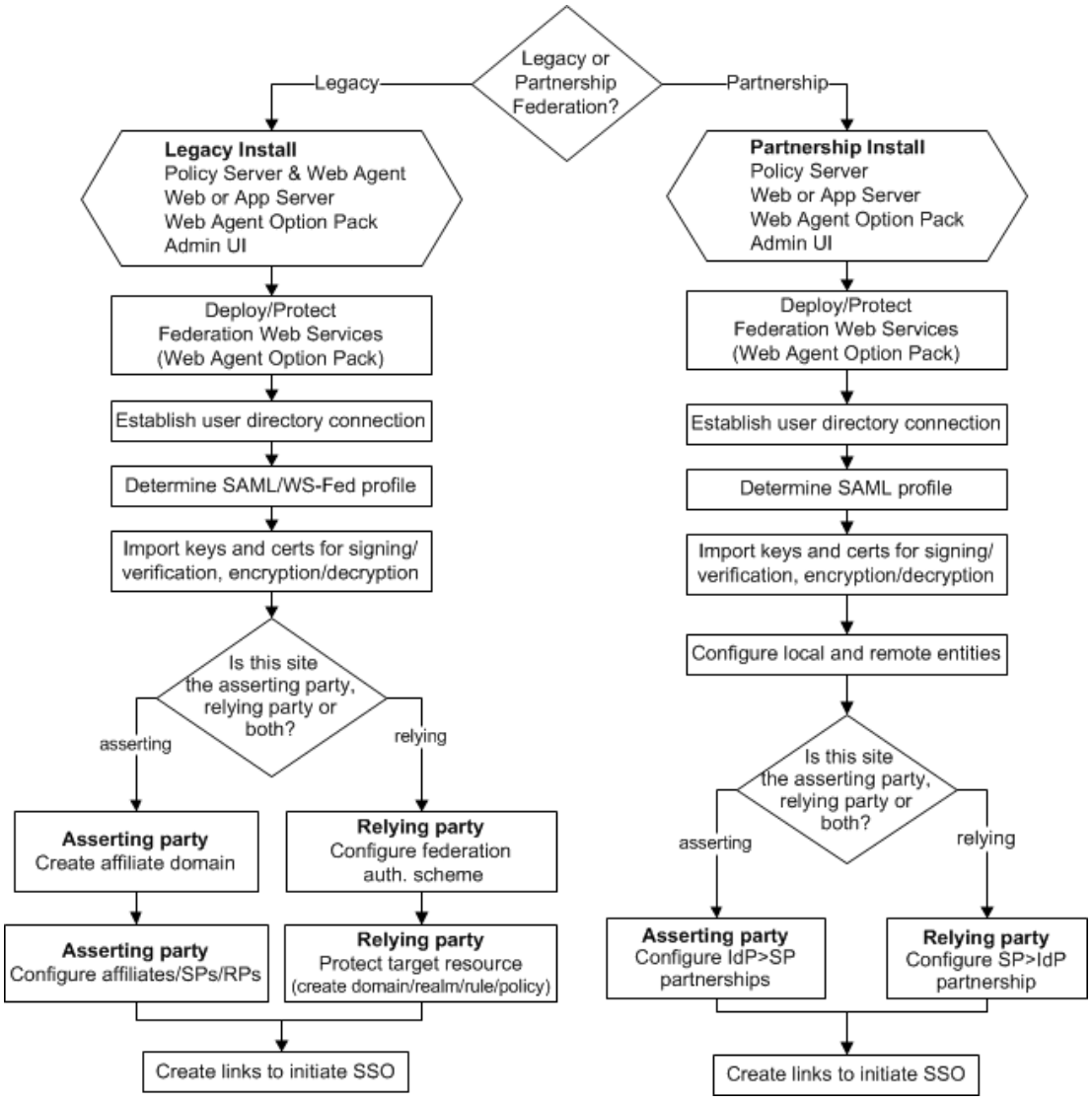
1. Setting up SAML and WS-Federation authentication schemes
   Enables you to configure redirection to the target application, lets you set up provisioning of user records, and define relying-party side attribute mapping.

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.



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

- **Partnership federation**
  Partnership Federation refers to partnership model of federation.

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

Comparing Federation and Web Access Management for Single Sign-on

This content compares Federation and Web Access Management (WAM) approaches for implementing single sign-on.

- Advantages of Federation and Web Access Management (see page 199)
- Deployments that Favor Federation (see page 200)
- Deployments that Favor Web Access Management (see page 200)

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 CA Single Sign-On 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.

CA Single Sign-On WAM single sign-on has the following advantages:

- Transactions are faster because there are fewer browser redirects.

- CA Single Sign-On 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.

- CA Single Sign-On 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 CA Single Sign-On 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.

Federation Web Services

This content discusses Federation Web Services (FWS).

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 Policy Server. The Federation Web Services and the 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 and partnership federation)

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 CA Single Sign-On session store. The SAML specification defines the assertion retrieval request and response behavior.

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

- **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 CA Single Sign-On 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 CA Single Sign-On 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 CA Single Sign-On 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 Profile

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

- **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 cookies to a browser.

- **Single Sign-on Service**
  Enables an Identity Provider to process a sign-on 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.

- **Sign-out Service**
  Implements processing of a single sign-out transaction by way of a sign-out servlet. An Identity Provider or a Resource Partner can initiate sign-out.

Implementing CA SiteMinder® SPS

The content in this section provides architectural and configuration information to consider when deploying CA Access Gateway. Use the Table of Contents to access the content.

CA SiteMinder® SPS Architecture Introduced

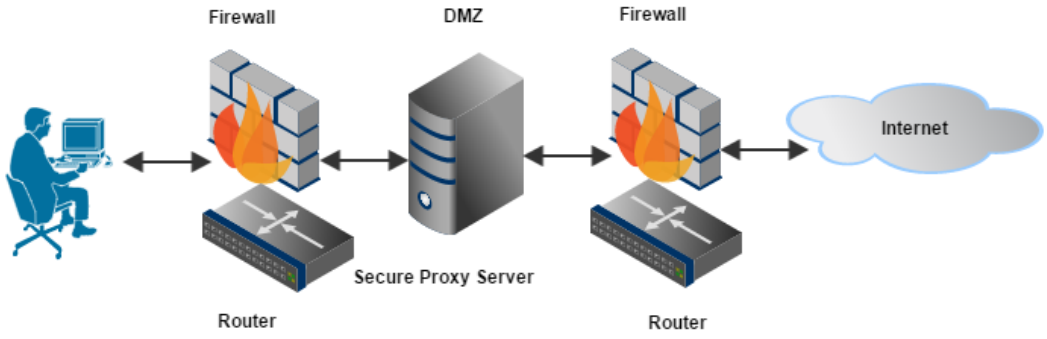
Contents

CA Access Gateway is a stand-alone server that provides a proxy-based solution for access control. CA Access Gateway employs a proxy engine that provides a network gateway for the enterprise and supports multiple session schemes that do not rely on traditional cookie-based technology.

Proxy Server Architecture

A traditional proxy server is located between a firewall and an internal network and provides caching of resources and security for the users on the internal network. Traditional proxy servers act as a proxy on behalf of a group of users for all resources on the Internet.

The following illustration shows a proxy server configuration. The proxy server caches frequently accessed resources so that requests for those resources are handled faster in the Demilitarized Zone (DMZ).



Traditional Reverse Proxy Server Architecture

A reverse proxy server represents one or more destination servers. A typical use of reverse proxy architecture provides:

- Cached resources
- Security
- SSL acceleration
- Load balancing

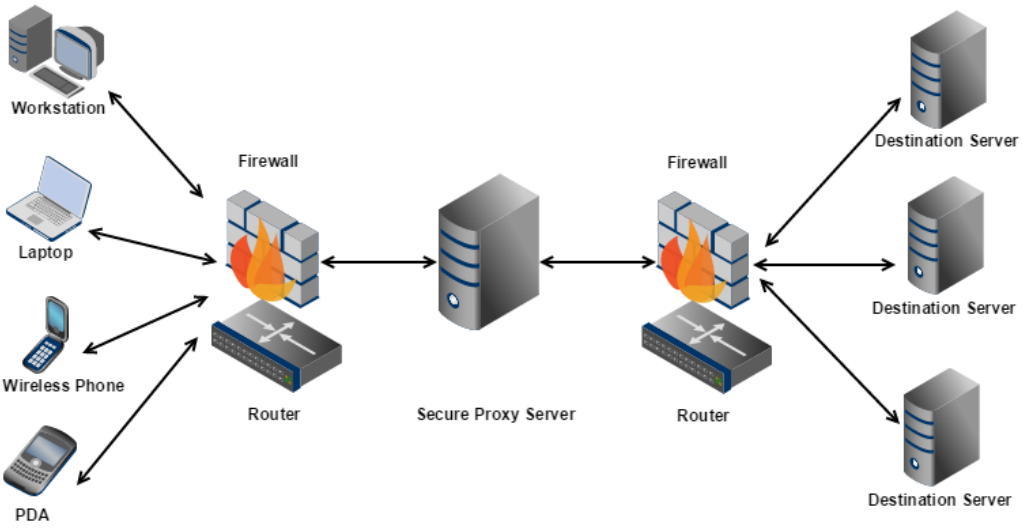
Rather than requesting a resource directly from a destination server, the reverse proxy server caches much of the content from the destination servers, providing ready access for users. This type of proxy server deployment is considered a reverse proxy, because the proxy is transparent to the user and works on behalf of the destination servers in the enterprise. Multiple reverse proxy servers can be used for load balancing and can also provide some SSL acceleration for HTTPS requests. A reverse proxy server also provides an additional layer of security by insulating destination servers that reside behind the DMZ.

CA Access Gateway Architecture

CA Access Gateway is not a traditional reverse proxy solution, because it does not provide resource caching. CA Access Gateway serves as a single gateway for access to enterprise resources, regardless of the method of network access.

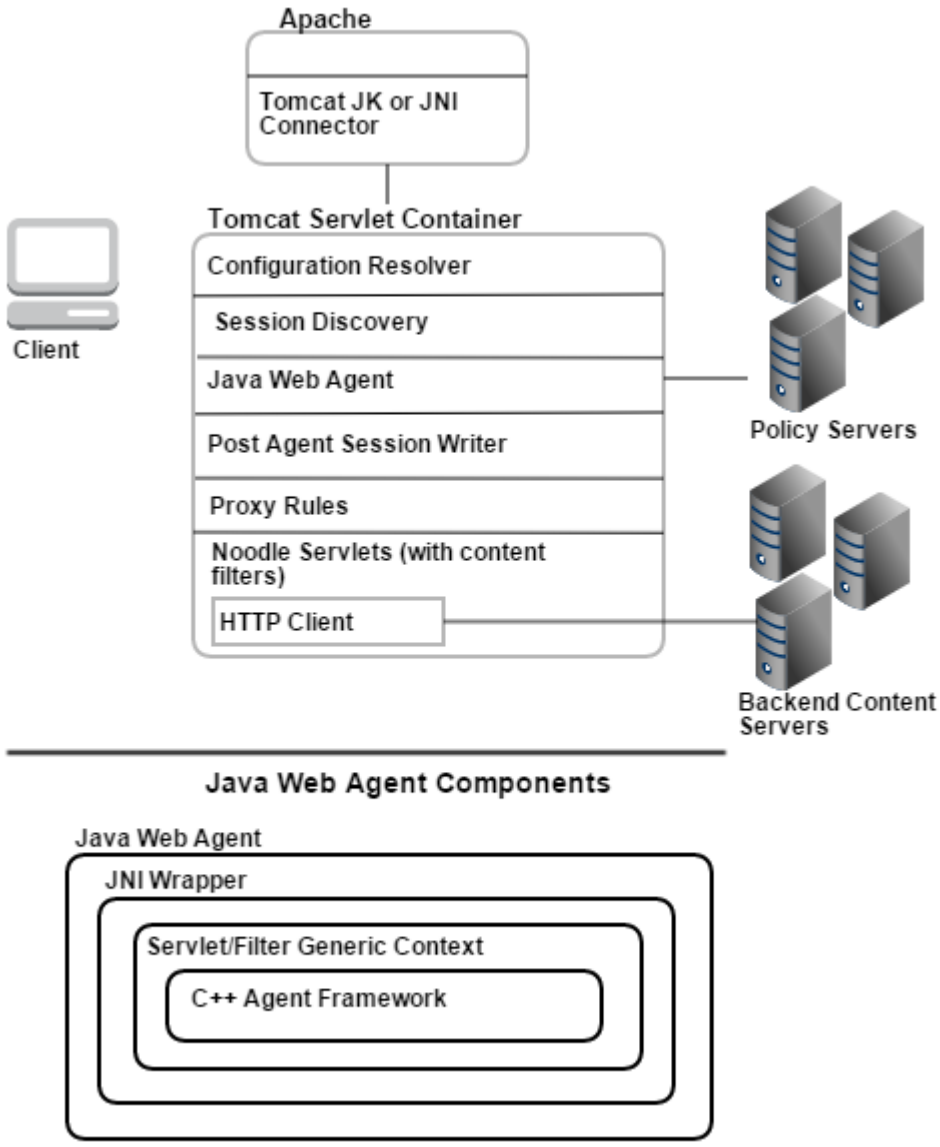
A set of configurable proxy rules determines how CA Access Gateway handles a user request. Users can access resources through multiple session schemes based on mapping between user agent types and virtual hosts. Requests can be routed to different destination servers based on the type of device being used to access the network.

The following illustration shows a configuration of CA Access Gateway. Users access CA Access Gateway using various devices. CA Access Gateway determines the session scheme to create based on the access device, and then forwards or redirects requests to the appropriate destination servers. Users are not aware that there is a reverse proxy server in the enterprise.



Components

A stand-alone CA Access Gateway consists of an HTTP listener (Apache) and a Tomcat servlet container, as shown in the following illustration:



CA Access Gateway architecture includes the following components:

- **Apache**
  CA Access Gateway uses the open source Apache Web server to act as the HTTP listener for incoming requests. An additional component, mod_jk acts as the Tomcat connector, which enables communication between the Apache Web server and Tomcat using the Apache JServ protocol (AJP).

- **Tomcat**
  The Tomcat server provides Tomcat servlet container for CA Access Gateway. The Tomcat initialization is customized so it does not allow deployment of any external applications or servlets. The standard Tomcat xml (server.xml) is not used for initialization. The components inside the Tomcat container of CA Access Gateway include the following:

- **Configuration Resolver ProxyBootstrap**
  The configuration resolver proxybootstrap is responsible for reading the CA Access Gateway configuration from the server.conf file and initializes the CA Access Gateway.

- **Session Discovery**
  The session discovery component analyzes the incoming requests for extracting CA Access Gateway session information. Depending on the user agent type and the virtual host being used, this component uses the appropriate session scheme for extracting the CA Access Gateway session information.
  If the request uses an existing CA Access Gateway session, this component uses the CA Access Gateway session identifier contained in the request to extract the corresponding CA Single Sign-on session from the in-memory session store. CA Access Gateway passes the CA Single Sign-on session to the Java Web Agent for session validation. If the request does not contain an existing CA Access Gateway session, this component passes the request on to the Java Web Agent for user authentication.

- **Java Web Agent**
  The Java Web Agent, together with the CA Single Sign-on Policy Server, authenticates and authorizes the user.

- **Post Agent Session Writer**
  The post Agent session writer performs additional processing for cookieless session schemes. After the Java Web Agent authenticates and authorizes the user and creates a CA Single Sign-on session, this component creates a CA Access Gateway session identifier. This identifier is attached to the CA Single Sign-on session created by the Java Web Agent.
  This session identifier is then maintained in the in-memory session store of CA Access Gateway. In addition to maintaining the session in the session store, this component transforms the URI. For example, the Post Agent Session Writer manipulates the URI for the simple_url session scheme.

- **Proxy Rules Servlet Filter**
  The proxy rules servlet filter loads the proxy rules from the proxyrules.xml file. Depending upon the incoming request and the proxy rule, the request is forwarded or redirected to the backend server. If the request is forwarded, an open source component Noodle is used.
  Any changes made to the proxy rules do not require a restart for the changes to take effect. The proxyrules are reloaded when there is any change in the proxyrules.xml file.

- **Noodle Servlet**
  The Noodle servlet forwards requests to the backend servers. Noodle also supports the use of proxy pre-filters which enable the request to be modified before sending the same to the backend server. Similarly support for proxy post-filters is also available which enables modification of the response received from the backend server before sending it back to the user client.

- **HTTP Client**
  The HTTP client sends requests to the backend server and receives responses from the backend server.

Product Features

CA Access Gateway offers the following features:

- **Access Control for HTTP and HTTPS Requests**
  CA Access Gateway allows you to control the flow of HTTP and HTTPS requests to and from destination servers using an embedded CA Single Sign-on web agent. In addition, CA Access Gateway is fully integrated with CA Single Sign-on to manage e-business transactions.

- **Single Sign-on**
  The embedded web agent in CA Access Gateway enables single sign-on (SSO) across an enterprise, including SSO with CA Single Sign-on Web agents that can be installed on destination servers within the enterprise.

- **Multiple Session Schemes**
  A session scheme is a method for maintaining the identity of a user after authentication. Core CA Single Sign-on products use cookies to maintain a session. CA Access Gateway, however, can maintain sessions based on SSL ID, mini-cookies, device IDs for handheld devices, URL rewriting, IP addresses, and schemes created using the Session Scheme API.

- **Session Storage**
  CA Access Gateway is equipped with an in-memory session store. The session store maintains session information. CA Access Gateway uses a token, such as a mini-cookie or SSL ID, to refer to the session information in the session store. Multiple session schemes and in-memory session storage enable CA Access Gateway to provide a solution for e-business management beyond computers, wireless devices such as PDAs and wireless phones.

- **Cookieless Single Sign-on**
  Some enterprises prefer solutions that do not use cookie technology. Because of the session schemes and the session store built into CA Access Gateway, it offers a solution to enterprises that want an alternative to cookie-based session management.

- **Intelligent Proxy Rules**
  Proxy rules allow you to configure different paths for fulfilling client requests from CA Access Gateway based on characteristics such as the requested virtual host or URI string. The proxy engine interprets a set of proxy rules to determine how to handle user requests.

- **Centralized Access Control Management**
  By providing a single gateway for network resources, CA Access Gateway separates the corporate network and centralizes access control.

- **Enterprise Class Architecture**
  CA Access Gateway is designed to be scalable, manageable, and extensible.

Product Limitations

The following conditions apply to CA Access Gateway:

- CA Access Gateway is not a plug-in to another Web server. CA Access Gateway is a fully supported, stand-alone server.

- CA Access Gateway does not support local content. The ability to place content on CA Access Gateway is not exposed, and CA Access Gateway does not support proxy rules for providing access to local content.

- CA Access Gateway does not support having the Web server on the same system as CA Access Gateway. If the two are set up on the same system, the Web server is accessible from the Internet. Security is not sure with this configuration.
- CA Access Gateway provides its own session storage. However, the session store has no public APIs for use as a general session server.
- In some enterprises that use CA Access Gateway, destination servers can also have CA Single Sign-on web agents or application server agents installed. When policies for CA Access Gateway agent are inconsistent with policies for the agent installed on the destination server, CA Access Gateway can pass responses back to the invoking client. Because CA Access Gateway does not provide warnings about inconsistencies in processing such policies, use care when setting up CA Single Sign-on policies in such environments.
- CA Access Gateway makes a new request to the destination server for every request that it receives. All caching directives are ignored.
- In the simple-url session scheme, CA Access Gateway does not rewrite URLs embedded in or resulting from JavaScript.
- The simple_url session scheme does not preserve the POST data when posting to a protected resource.

CA Access Gateway in an Enterprise

Enterprises that provide access to network resources for employees, customers, and partners face a number of challenges, including:

- Directing requests to appropriate services
- Verifying user identities and establishing entitlements
- Maintaining sessions for authorized users
- Providing centralized access control configuration
- Supporting multiple device types
- Employing flexible and secure architectures

CA Single Sign-on provides solutions to many of these challenges, including authentication and authorization of users, and a complex engine for evaluating user entitlements. CA Access Gateway further expands the benefits of core Policy Server and Web Agent functionality by providing a reverse proxy solution.

This reverse proxy solution adds the following capabilities:

- Inter-operability with existing CA Single Sign-on Web Agents
- Cookieless single sign-on and sessions storage
- Centralized configuration through proxy rules

- Multiple options for maintaining sessions
- Multiple device support

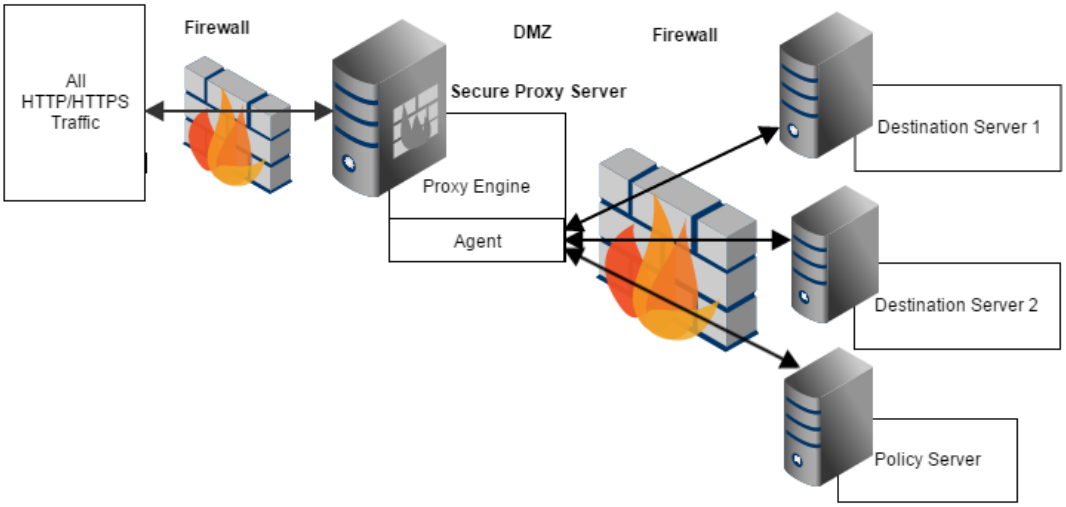
You can deploy CA Access Gateway in an enterprise to serve the following functions:

- Act as a centralized access control filter
- Support cookieless session schemes
- Support extranet access control

CA Access Gateway as a Centralized Access Control Filter

To limit access to destination servers and provide a central entry point to the network, CA Access Gateway can be placed in front of all destination servers in the enterprise. HTTP or HTTPS requests that come into the enterprise can be filtered through CA Access Gateway, and forwarded to the appropriate destination server for fulfillment.

The following illustration shows how CA Access Gateway handles all HTTP and HTTPS requests.



Destination servers that contain content do not require CA Single Sign-on Web Agents. The only network element that resides behind the first firewall is CA Access Gateway. All users must be authenticated and authorized by CA Single Sign-on residing behind the second firewall. The destination servers provide content after CA Single Sign-on and CA Access Gateway verify user entitlements.

This deployment provides the following benefits:

- Centralizes configuration through proxy rules
  CA Access Gateway uses proxy rules defined in XML configuration files to establish how CA Access Gateway handles requests. Proxy rules can be based on:
  - Host name

- URI substring
- HTTP header
- CA Single Sign-on header
- Regular Expressions based on URI

In addition, the conditions for proxy rules can be nested to create rules that incorporate multiple conditions.

- Directs requests to appropriate services
  All HTTP and HTTPS traffic passes through CA Access Gateway. Based on the proxy rules established for CA Access Gateway, requests are forwarded to the appropriate destination servers for fulfillment.

- Verifies user identities and establishes entitlements
  CA Access Gateway uses the built-in web agent to communicate with CA Single Sign-on and perform authentication and authorization of requests.

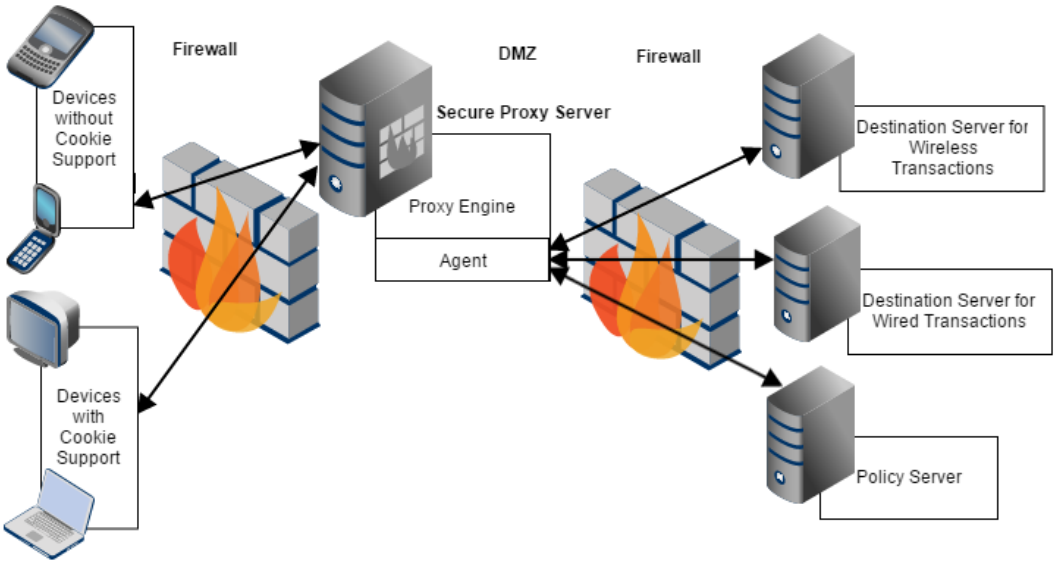
CA Access Gateway Support for Cookieless Sessions

Most solutions use cookie technology. However, when accessing resources over HTTP or HTTPS, some enterprises want alternatives for establishing and maintaining a user session and provide single sign-on with a cookieless solution.

CA Access Gateway provides an in-memory session store and allows the use of any of the following cookieless session schemes:

- Mini-cookies (uses small cookies in place of standard CA Single Sign-on cookies)
- SSL ID
- Device ID
- Simple URL Rewriting
- IP Address

The following illustration shows a deployment in which CA Access Gateway provides a combination of standard sessions using cookies and sessions without cookies:



The deployment shown in the previous illustration provides the following benefits:

- Supports multiple device types
  Through a set of proxy rules, CA Access Gateway forwards, or redirects, requests based on the type of device issuing the requests. For example, all initial requests can be directed at CA Access Gateway, which forwards requests to destination servers based on device types. Browser requests can be redirected to destination servers, and CA Access Gateway handles wireless requests.

- Maintains sessions for authorized users
  Both standard CA Single Sign-on cookies and cookieless session schemes are employed for maintaining user sessions. Session schemes are assigned based on user agent type for each virtual host. For example, all users accessing the network through web browsers are assigned to a standard cookie session scheme. Users accessing resource through a wireless telephone are assigned to a device ID session scheme.

- Provides cookieless single sign-on and session storage
  Through an in-memory session store and the support of multiple session schemes, CA Access Gateway provides alternatives to cookie-based sessions. CA Access Gateway maintains session information in the session store and returns a token. This token is exchanged with all transactions, allowing CA Access Gateway to match the token to the session information captured in the session store.

- Offers multiple options for maintaining sessions

Cookieless Session Scheme in a Federation Environment

CA Access Gateway, with its built-in handling of cookieless session schemes, enables it to be deployed in environments where the user agent, such as a wireless device, does not support traditional CA Single Sign-on cookies.

If you deploy CA Access Gateway in a CA Single Sign-on federation security services environment, the following process is enforced when a user request is received:

1. CA Access Gateway receives a request for a federated resource. The request is directed to the Federation Web Services (FWS) application at the site producing assertions.

2. CA Access Gateway verifies if cookieless federation is enabled for the virtual host requesting the redirect.

3. If a cookieless scheme is being used, CA Access Gateway removes the session key (SMSESSION cookie) for the current session.

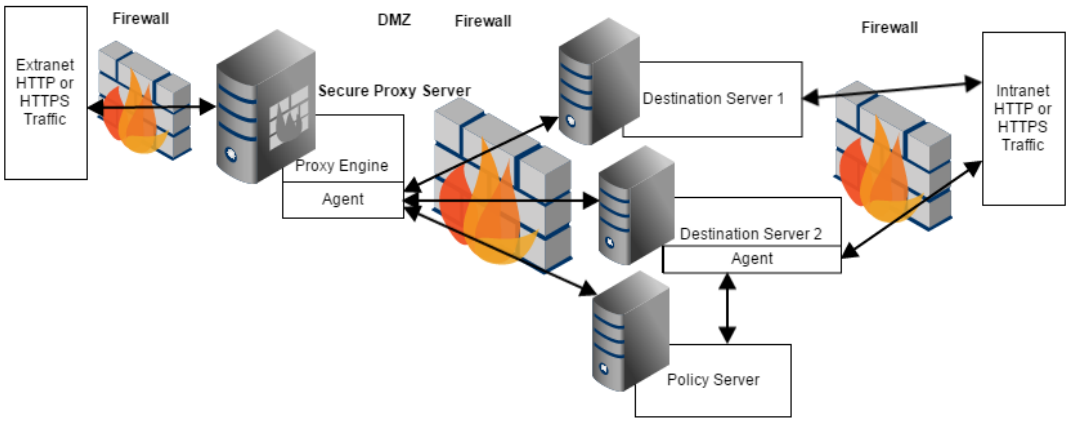
4. CA Access Gateway sends the user to the link provided by the FWS redirect.

If CA Access Gateway is using a rewritable session scheme such as simple_url session scheme, CA Access Gateway rewrites the redirect response to include the session key information in the redirected URL.

CA Access Gateway Support for Extranet Access Control

Another deployment of CA Access Gateway provides access control for external users, but allows direct access to destination servers for internal users. If a destination server provides access to secure applications for individuals within the enterprise, a standard CA Single Sign-on Web Agent can be installed on the destination server to provide access control. Users who are properly authenticated through CA Access Gateway can use single sign-on.

The following illustration shows an example of an extranet network deployment.



This deployment provides the following benefits:

- Directs requests from extranet sources
  All extranet traffic passes through CA Access Gateway and is forwarded to the appropriate destination server once users are authenticated and authorized for requested resources.

- Employs flexible architectures
  All information is located behind multiple firewalls to protect from extranet attacks. Information that is appropriate for intranet users does not incur the overhead of agent to CA Single Sign-on communication. CA Single Sign-on can still protect sensitive resources, however.

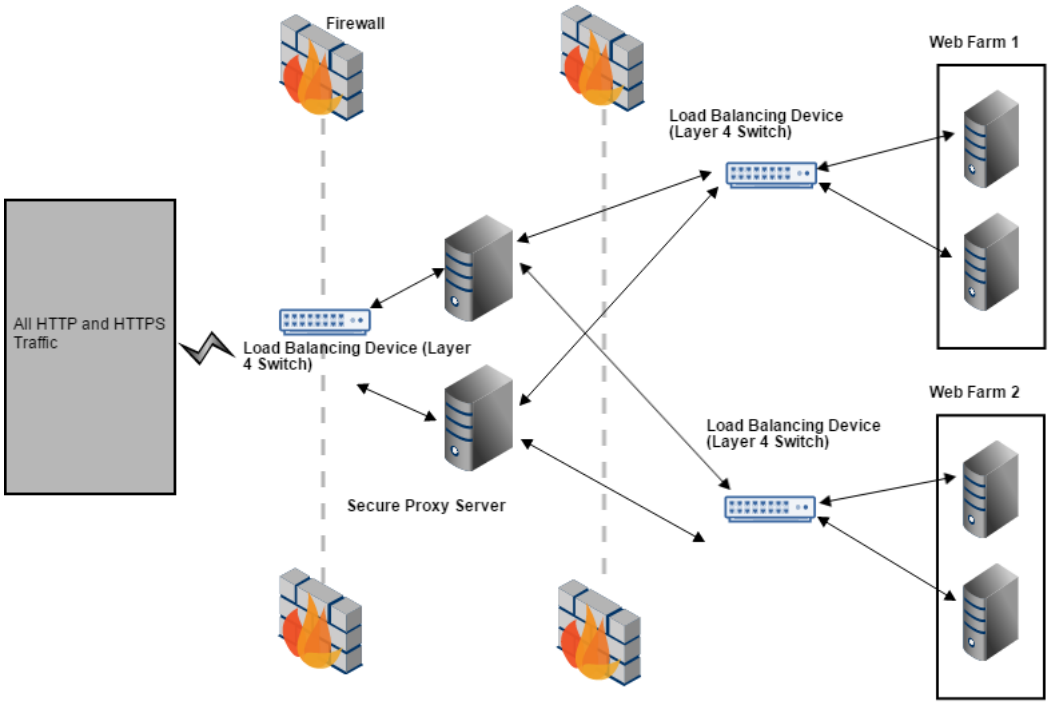
- Provides interoperability between Web Agents
  CA Access Gateway and intranet Web Agents use the same Policy Server and provide single sign-on for authorized extranet users on all destination servers.

CA SiteMinder® SPS in an Enterprise

Contents

CA Access Gateway uses reverse proxy architecture to enable access control, single sign-on, and SSL acceleration. It does not provide the content caching and some other features provided by traditional reverse proxy servers. The CA Access Gateway is intended to be an addition to enterprise architecture, rather than a replacement for other proxy technologies. As such, the CA Access Gateway can be placed in clusters with load balancing devices and caching devices on either side of the clusters.

The following illustration shows how the CA Access Gateway can be deployed in a network to work in conjunction with load balancing devices:



Note: In addition to load balancing devices, caching devices can be placed on either side of the CA Access Gateway cluster.

Sticky-Bit Load Balancing

When using the cookieless session schemes supported by the CA Access Gateway, session information for users who access resources through CA Access Gateway is maintained in an in-memory session store. Because the session information is maintained at the CA Access Gateway where a user is first authenticated, the same CA Access Gateway should be used for all the user requests in a single session. When implemented in clusters, the CA Access Gateway must be used in conjunction with sticky-bit load balancers to provide a consistent connection to the same CA Access Gateway, enabling single sign-on when using session schemes other than the traditional CA Single Sign-on cookie session scheme.

To deploy the CA Access Gateway using cookieless session schemes the following must be considered:

- In most deployments, the CA Access Gateway is deployed in clusters, with several servers sharing the load of incoming requests. The load balancing is handled by load balancer devices. These devices must have sticky bit capability to maintain single sign-on.
  Sticky bit load balancers ensure that once a user's session is established with a specific CA Access Gateway in a cluster, that CA Access Gateway services all of the user's requests. This capability is required because the CA Access Gateway maintains session information for cookie-less sessions in active memory. If a user's request is not handled using sticky bit technology, the user will be charged for new credentials each time a request is fulfilled by a different CA Access Gateway in the cluster of servers.

- When configuring the settings for the CA Access Gateway, the default virtual host defined in the server.conf file of the CA Access Gateway must be defined using the name and IP address of the load balancing device.

- The load balancing device must be configured as the point of entry to the CA Access Gateway.

- The load balancing device must point to the cluster of CA Access Gateways.

- The httpd.conf file, located in the *sps_home*/secure-proxy/httpd/conf directory, must be modified so that the value of the ServerName directive is set to the name of the load balancing device, not the system on which you installed the CA Access Gateway.

- If using SSL, a certificate must be issued to the load balancer, not the CA Access Gateway.

- The system or systems on which you install the CA Access Gateway must have approximately 1K of memory for each simultaneous user session that will be maintained in the in-memory session store. For example, if a single system must maintain 1000 concurrent sessions, the system must have 1 megabyte of RAM available for this purpose.

Proxying to Trusted Sites vs. Non-Trusted Sites

The CA Access Gateway is assumed to proxy for trusted sites within the enterprise. As part of a proxy transaction, CA Single Sign-on generated HTTP header variables and any variables generated by CA Single Sign-on responses are forwarded along with each HTTP and HTTPS request. These responses can be used by other enterprise applications.

Important! If you employ the CA Access Gateway in transactions that proxy for content on non-trusted sites, the headers that accompany the transaction will also be forwarded to the non-trusted sites. We recommend using the CA Access Gateway to proxy for destination servers trusted by your enterprise.

Configuring Virtual Hosts

You can configure the CA Access Gateway with multiple hosts and act as a virtual host for one or more hostnames.

Follow these steps:

1. Edit the <VirtualHost> parameters of the server.conf file to configure the CA Access Gateway to act as a virtual host for one or more hostnames.

2. Edit the configuration file for the embedded Apache Web server.

Edit the Apache Configuration File To Handle Multiple Virtual Hosts

When you are running multiple virtual hosts in the same operating environment with the CA Access Gateway, and transactions run in this environment, update the Apache configuration file (httpd.conf). This file is located in the directory *sps_home*\secure-proxy\httpd\conf. If SSL is enabled for the web server, also make the same updates to the httpd-ssl.conf file, which is located in the *sps_home* \secure-proxy\httpd\conf\extra directory. The updates vary depending on whether your operating environment is based on IPv4 or IPv6.

To update the httpd.conf file, and optionally the httpd-ssl.conf file, to handle multiple virtual hosts

- For IPv4 environments, add the following LISTEN directive:
  LISTEN 127.0.0.1:<*_port*>

- For IPv6 environments, add the following LISTEN directive:
  LISTEN [::1]:<*_port*>

- For dual stack environments with IPv4 and IPv6 supports, add the following LISTEN directives:
  LISTEN 127.0.0.1:<*_port*>
  LISTEN [::1]:<*_port*>

In addition, update the loopback address entry in the hosts file so that the new host name is added, as follows:

- IPV4: 127.0.0.1
- IPV6: [::1]

The hosts file is usually located on Windows in C:\WINDOWS\system32\drivers\hosts. On UNIX, the hosts file is usually in /etc/hosts.

Implementing Session Scheme Mappings for Multiple Virtual Hosts

If you want to configure the CA Access Gateway to recognize multiple user agent types, assign different session scheme mappings for those user agents based on virtual hosts, you must follow these steps:

1. Configure session schemes, or verify the configuration of the schemes included with the CA Access Gateway.
2. Define user agent types in the server.conf file.
3. Create a section for each virtual host in the server.conf file that defines any directives that differ from default settings (refer to Overriding Default Values for a Virtual Host).
4. Define session scheme mappings for each virtual host.

The following excerpts from a server.conf file provide an example where a user agent type has been defined for Internet Explorer (IE) browser users. IE users will be mapped to use session schemes other than the default session scheme defined for a virtual host. The following example shows the session schemes defined in the server.conf file.

```
#Session Schemes
<SessionScheme name="default">
    class="com.netegrity.proxy.session.SessionCookieScheme"
    accepts_smsession_cookies="true"
</SessionScheme>
<SessionScheme name="ssl_id">
    class="com.netegrity.proxy.session.SSLIdSessionScheme"
    accepts_smsession_cookies="false"
</SessionScheme>
<SessionScheme name="simple_url">
    class="com.netegrity.proxy.session.SimpleURLSessionScheme"
    accepts_smsession_cookies="false"
</SessionScheme>
<SessionScheme name="minicookie">
    class="com.netegrity.proxy.session.MiniCookieSessionScheme"
    accepts_smsession_cookies="false"
    cookie_name="MiniMe"
</SessionScheme>
```

The following example shows the definition of the IE user agent type. This user agent type will be referenced when defining session scheme mappings later in the server.conf file.

```
# TO-DO: Define Any User Agents, if you want to
# use a different session scheme based on
# the type of client accessing the server.
#
# NOTE:  UserAgent matching is done in the order
# in which the user agents are defined in this file.
 <UserAgent name="IE">
     User-Agent="MSIE"
 </UserAgent>
# <UserAgent name="NS">
#     User-Agent=some other regular expression
# </UserAgent>
```

The preceding example shows that the default session scheme specified in the defaultsessionscheme directive is mini-cookie. This session scheme will be used for all transactions unless another session scheme is explicitly included in a session scheme mapping, or another scheme overrides the default session scheme in the definition of a virtual host.

The <VirtualHostDefaults> directive shows the session scheme mapping for the IE user agent type that was defined in <UserAgent name="IE">. This mapping indicates that for all virtual hosts using default session scheme mappings, IE browser users' sessions will be maintained using the simple URL rewriting sessions scheme.

```
<VirtualHostDefaults>
    # Service Dispatcher
    <ServiceDispatcher>
        class="com.netegrity.proxy.service.SmProxyRules"
        rules_file="conf\proxyrules.xml"
    </ServiceDispatcher>
    # default session scheme
    defaultsessionscheme="minicookie"
    #TO-DO:  Define any session scheme mappings
    <SessionSchemeMappings>
       #    user_agent_name=session_scheme_name
           IE="simple_url"
       #    NS=simple_url
    </SessionSchemeMappings>
```

The Virtual Host directives show the server name and IP address for the default virtual host configured for the CA Access Gateway.

```
# Default Virtual Host
<VirtualHost name="default">
    hostnames="server1, server1.company.com"
    addresses="192.168.1.10"

    #The defaults can be overriden
    #not only for the Virtual Host
    #but for the WebAgent for that
    #virtual host as well
```

```
    #<WebAgent>
    #</WebAgent>

</VirtualHost>
```

The Virtual Host directive for additional virtual host shows the specific default virtual host settings that will be overridden for the server2 virtual host. Notice that these overrides include new session scheme mappings. The default scheme for server2 is default. In Session Scheme directive the default is defined as the traditional CA Single Sign-on cookies session scheme. Further, the session scheme mapping for IE users in Virtual Host directives is also mapped to the default scheme. Therefore, the CA Access Gateway will use CA Single Sign-on cookies session scheme to maintain sessions for all users who access server2.

```
# Additional Virtual Host
<VirtualHost name="host2">
       requestblocksize="4"
    responseblocksize="4"
    hostnames="server2, server2.company.com"
    #addresses="192.168.1.15"
    # default session scheme
    defaultsessionscheme="default"

    #TO-DO:  Define any session scheme mappings
    <SessionSchemeMappings>
       #user_agent_name=session_scheme_name
           IE="default"
    </SessionSchemeMappings>

    #<WebAgent>
    #</WebAgent>
</VirtualHost>
```

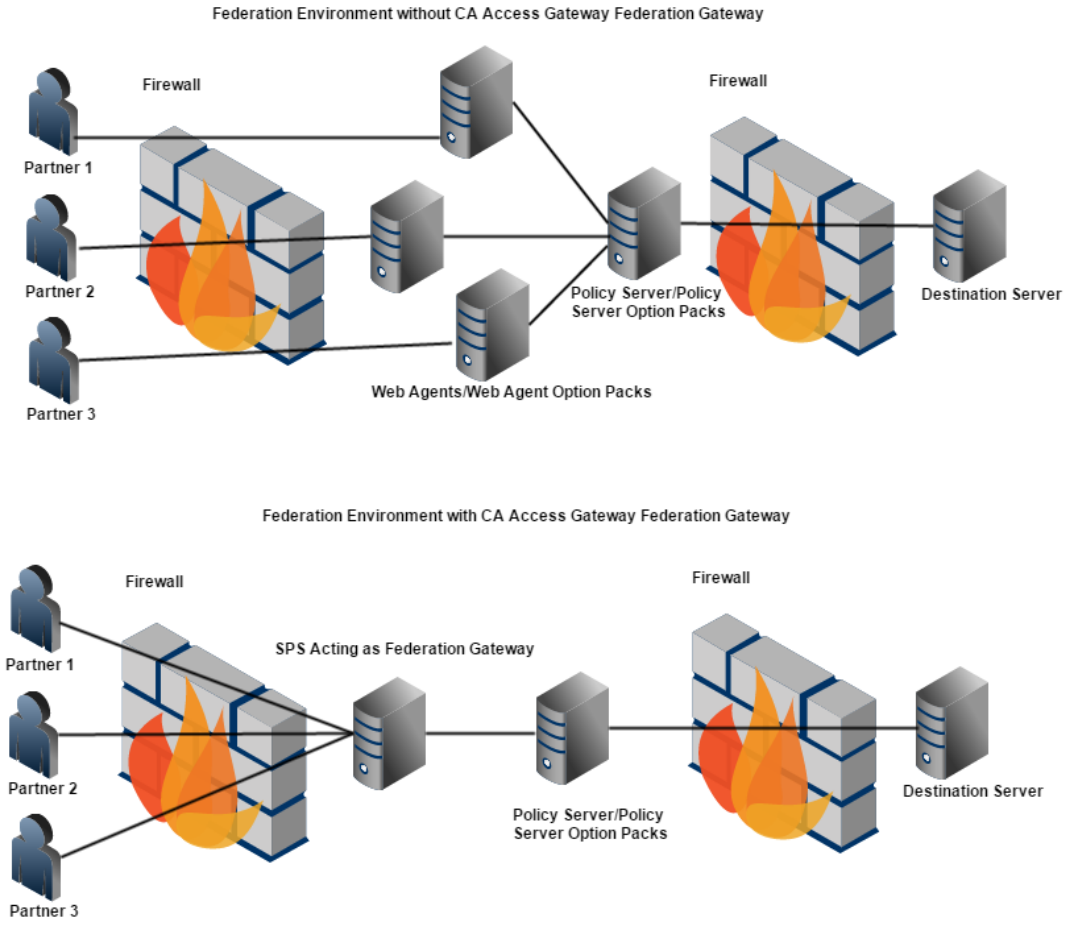
Using CA SiteMinder® SPS as a Federation Gateway

Contents

CA Access Gateway federation gateway simplifies the configuration involved in a federated environment. Typically, you have a federated environment where partners are communicating through many web servers. Each web server requires that you install and configure the Web Agent and the Web Agent Option Pack.

If you enable CA Access Gateway as a federation gateway, the number of components that you have to install and set-up is reduced. The CA Access Gateway federation gateway has the standard embedded components of CA Access Gateway and the Federation Web Services application provided by the Web Agent Option Pack.

The following illustration shows the difference with or without the CA Access Gateway federation gateway.



Prerequisites for Using the Federation Gateway

Before you set up CA Access Gateway as a federation gateway, consider the following:

- The CA Single Sign-on environment must be configured according to the information in the *CA Single Sign-on Federation Security Services Guide*. Verify that the Policy Server-side components for federation are configured.
- Install CA Access Gateway and enable the enablefederationgateway setting when prompted.

- In the CA Single Sign-on Administrative UI, be sure to define the host information (server and port number) for the CA Access Gateway system that generates assertions. The CA Access Gateway host is defined in the Server field of the appropriate properties dialog for the federated partner you are specifying.

Configuring the CA Access Gateway Federation Gateway

The CA Access Gateway federation gateway can sit at the producer site and consumer site.

The overall configuration process for the CA Access Gateway federation gateway is as follows:

1. Install CA Access Gateway.
2. Run the configuration wizard and enable the enablefederationgateway option.
3. Specify the general server settings in the server.conf file. Though there are defaults for most of the server.conf settings, you may want to modify such settings as session schemes or virtual host settings.
4. Define proxy rules in the proxyrules.xml file so that requests are directed to the backend servers.
   At the enterprise producing assertions, federation requests are forwarded to the Tomcat server embedded in CA Access Gateway. The Tomcat server hosts the FWS application.
   At the enterprise consuming assertions, you need to define a proxy rule that forwards requests to the destination server after the user is permitted access to the target resource.
5. (Optional) You can modify the Apache web server file (httpd.conf).

Limitations of the CA Access Gateway Federation Gateway

Note the following limitations when using the CA Access Gateway federation gateway:

- The prefilters and postfilters (both built-in and custom-configured) do not execute when federation resources are being requested. For non-federated requests that are fired for the default context, these filters execute as usual.
- Proxy rules do not execute when federated resources are being requested. For non-federated requests that are fired for the default context, these rules execute as usual.

Using CA Access Gateway with FWS

For use cases on how to use CA Access Gateway with FWS, see Federation Use Cases and Solutions Common to SAML and WS-Federation (see page 154).

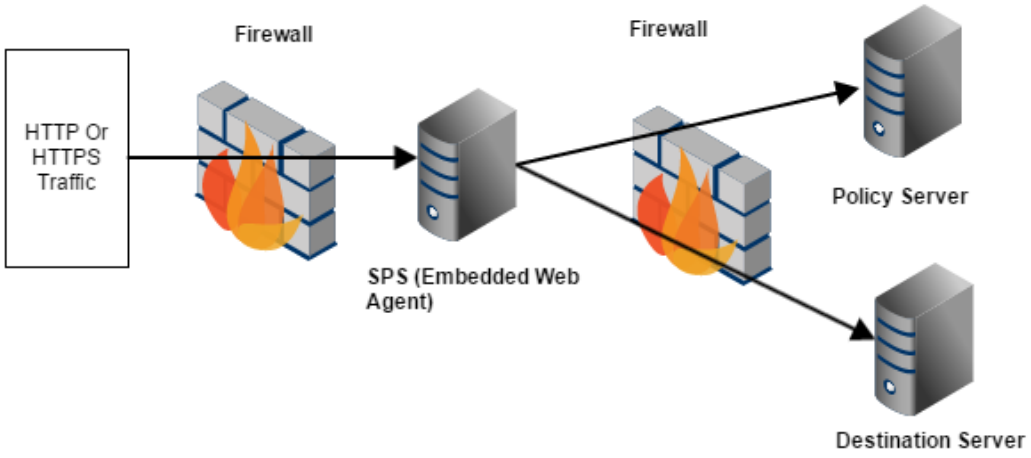
Using CA SiteMinder® SPS as a Web Agent Replacement

Contents

To provide federated single sign-on, CA Access Gateway may be used as a substitute for the Web Agent and Web Agent Option Pack. CA Access Gateway provides the Federation Web Services (FWS) application, which is a collection of servlets packaged as a Web application. This application provides much of the CA Single Sign-on federation functionality.

Knowledge of FWS is required for anyone configuring CA Access Gateway in a federated environment.

The following figure shows an environment where CA Access Gateway replaces a CA Single Sign-on Web Agent.



Important! If you choose to use CA Access Gateway in place of the Web Agent for a federated environment, the Web Agent Option Pack requires a dedicated web server and servlet engine.

Verify Prerequisites

Before you configure CA Access Gateway for use in a federation environment, consider the following:

- The CA Single Sign-on environment must be configured according to the information in the *CA Single Sign-on Federation Security Services Guide*. We recommend that you configure a Federation environment with a standard Web Agent to confirm that Federation Security Services is configured properly.

- In the CA Single Sign-on Administrative UI, define the host information (server and port number) for the CA Access Gateway system that generates assertions. The CA Access Gateway host is defined in the Server field of the appropriate properties dialog for the federated partner you are specifying.

Configure CA Access Gateway

The configuration process for CA Access Gateway to operate in a federated environment is similar to the standard CA Access Gateway configuration process.

The overall configuration process for the CA Access Gateway federation gateway is as follows:

1. Install CA Access Gateway.

2. Run the configuration wizard.

3. Specify the general server settings in the server.conf file. Though there are defaults for most of the server.conf settings, you can modify settings such as logging, session schemes, or virtual host settings.

4. At the enterprise producing assertions, define a proxy rule that forwards requests to the backend server hosting FWS. At the side consuming assertions, there must be a rule that forwards requests to the destination server after the user is permitted access to the target resource.

5. (Optional) If you want to configure virtual hosts for CA Access Gateway, you can modify the Apache web server file (httpd.conf), for example,

Using CA SiteMinder® SPS in Cookieless Federation

Certain devices or environments cannot use cookies to establish user session and provide single sign-on.

One type of session scheme you can use in a federated environment is a cookieless scheme. The cookieless federation scheme is used to establish single sign-on. Verify that FWS-generated cookies (session and attribute) are not sent back to clients using mobile devices that do not support cookies.

Cookieless Federation at the Producing Site

At the site producing assertions, the process for a cookieless transaction is as follows:

1. CA Access Gateway verifies if cookieless federation is enabled for the virtual host requesting the redirect.

2. CA Access Gateway verifies if the session scheme is a rewritable scheme, such as the simple_url scheme.

3. If the scheme is rewritable, CA Access Gateway determines whether a session key has been created for the session and if this key is available to use.

4. CA Access Gateway checks to see if the Location header in the HTTP response meets one of the following conditions:

   - It is being rewritten.

   - It is the same as the host of the request.

5. CA Access Gateway rewrites the redirect response to include the session key information in the redirected URL.

Cookieless Federation at the Consuming Site

At the site consuming assertions, if cookieless federation is enabled, CA Access Gateway replacing the Web Agent processes redirects using SAML authentication at the backend server.

In a cookieless federation, CA Access Gateway processes the request as follows:

1. CA Access Gateway receives a request from cookieless device, such as a mobile phone.

2. CA Access Gateway verifies if the cookieless federation is enabled for the virtual host requesting the redirect.

3. CA Access Gateway then checks to see if the following conditions have been met:

   - The response from the backend server is a redirect.

   - The response contains an SMSESSION cookie.

   If these two conditions are met at the same time, it indicates that a SAML authentication has occurred at the backend server from the FWS application.

4. CA Access Gateway retrieves the session scheme being used.

5. CA Access Gateway creates an associated cookieless session and adds the session information to its session store.

6. If the session scheme is rewritable, such as a simple URL session scheme, CA Access Gateway rewrites the location header with the session key.

7. If CA Access Gateway determines that a cookieless federated session conversion has occurred, CA Access Gateway deletes the SMSESSION cookie from the response going to the browser.

8. CA Access Gateway then checks to see if attribute cookies should also be deleted. It does this by checking the deleteallcookiesforfed parameter. If this parameter is set to yes, CA Access Gateway deletes all the other cookies from the response going to the browser.

Enable Cookieless Federation at the Consuming Side

When CA Access Gateway replaces the Web Agent at the side consuming assertions, the cookieless federation parameters are enabled for any cookieless session scheme implemented by CA Access Gateway.

Follow these steps:

1. Open noodle.properties file from *sps_home*/secure-proxy/Tomcat/properties.
2. Remove the '#' from the following two lines, and save the file.
   - filter._cookielessfederation_.class=org.tigris.noodle.filters.CookielessFedFilter
   - filter._cookielessfederation_.order=1

   The settings are saved.
3. Open the server.conf file located at *sps_home*/secure-proxy/proxy-engine/conf.
4. Add the following code to the virtual host section for the virtual host that is serving the FWS:
   a. cookielessfederation="yes"
5. Save the file.
   CA Access Gateway is configured for cookieless federation at the consuming partner.

Deploy and Manage CA Access Gateway for NetScaler SDX

Valid for SDX Version 10.5 MR3

CA Access Gateway software is delivered as an appliance for deployment on NetScaler SDX. The CA Access Gateway for NetScaler SDX appliance contains the CA Access Gateway software and the operating system files that are required to function properly on the SDX platform.

Use the NetScaler SDX User Interface (UI) to deploy the appliance. After the deployment, configure the proxy rules and any other configurations that are required for your environment using the CA Access Gateway Administrative UI.

In this article, instance refers to the CA Access Gateway instance.

Note: For more information about the features of CA Access Gateway, see the CA Access Gateway Architecture Introduced page.

This release of the product is not localized.

This article contains the following sections:

Verify the Prerequisites

Make sure that the following prerequisites are met before deploying the appliance:

- A CA Single Sign-On (CA SSO) Policy Server is running.
- Gather the following information from the CA Single Sign-On policy administrator before you proceed with the deployment:
  - Trusted Host Name
  - Agent Name
  - Agent Configuration Object (ACO)
  - Host Configuration Object (HCO)
  - FIPS Mode of the CA Single Sign-On Policy Server
  - Policy Server IP address
  - CA Single Sign-On administrator credentials\
  - Master Key

Perform the following steps to download the .xva file from the CA Support website:

1. Go to http://ca.com/support.
2. Click CA Support Online.
3. Click Login.
4. Enter your CA Support Online credentials.
5. Mouse over the Download Center tile and click Download Products.
6. In Select a Product, type CA Access Gateway for Netscaler SDX - MULTI-PLATFORM.

The release and Gen level are automatically loaded.

7. Click Go.
8. Click Download.
9. Select a download method that you want to download the product.

Review the Known Issue

Review the following known issue before you deploy the instance:

Cannot Stop the Instance after Provisioning

After provisioning the instance, you are not able to stop the instance for the first time because the Apache process is still running. Reboot the virtual machine that has the instance to stop the Apache process.

Rebooting the virtual machine to stop the instance is required only once after provisioning the instance.

Upload the .xva file to the NetScaler SDX User Interface

Upload the .xva file that is obtained from the CA Support website to the NetScaler SDX User Interface (UI). Uploading the .xva file is a one-time process.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. Click CA Access Gateway, Software Images.
4. Click the XVA Files tab.
5. Click Upload.
6. Click Browse to locate the CA Access Gateway.xva file from your local computer.
7. Click Upload.

Provision a CA Access Gateway Instance

To use the capabilities of CA Access Gateway, provision the instance. CA Access Gateway is configured at the time of provisioning the instance.

Use the credentials of a CA Single Sign-On administrator with super user rights while provisioning the instance.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. In the left pane, click CA Access Gateway, Instances.
4. Click Add.
5. Specify the following parameters in the Provision CA Access Gateway section:
   - **Name**
     Defines the name of the virtual image on which the instance must be provisioned.
   - **Domain Name**
     Defines the domain name of the host on which the instance must be provisioned.
   - **XVA File**
     Identifies the .xva file that is obtained from the CA Support website to install a CA Access Gateway instance.
6. Specify the following parameters In the Initial Settings section:
   - **Trusted Host Name**

     Defines the trusted host name that must be used to identify the instance.
   - **Agent Name**

     Defines the Agent name that must be associated with the instance.
   - **Agent Configuration Object Name**

     Defines the name of the Agent Configuration Object (ACO) that must be associated with the instance.
   - **Host Configuration Object Name**

Defines the name of the Host Configuration Object (HCO) that must be associated with the instance.

- **Fully Qualified CA Access Gateway Server Host Name**

  Defines a fully qualified host name of the CA Access Gateway server in the following format:

  hostname.companyname.com

- **FIPS Mode**

  Specifies the FIPS mode of the Policy Server.

  Values: FIPS-Compatibility Mode, FIPS-Only Mode, FIPS-Migration Mode

  Default: FIPS-Compatibility Mode

- **Policy Server IP Address**

  Defines the IP address of the Policy Server where the instance must be registered. If the Policy Server is behind a firewall, then specify the Policy Server IP address with the port number.

- **Administrator Name**

  Defines the name of the CA Single Sign-On administrator that is already defined in the Policy Server. This administrator must have the privileges to create a trusted host and manage domain objects.

- **Administrator Password**

  Defines the password corresponding to the administrator name.

- **Master Key**

  Defines the master encryption key that is entered during the Policy Server configuration.

- **Enable Shared Secret Rollover**

- Specifies that the shared secret rollover is enabled in the Policy Server.

  Default: Not selected

- **Enable Web Agent**

  Specifies whether you want the instance to act as a Web Agent.

  Default: Not selected

- **Enable Federation Gateway**

Specifies whether you want the instance to act as a Federation Gateway.
Default: Not selected

7. Specify the following parameters In the Management Network section:

   - **IP Address**

     Defines the IP address of the virtual machine on which the instance is running.

     Note: The NetScaler SDX UI does not support IPv6 format for the virtual machine.

   - **Gateway**

     Defines the gateway of the virtual machine.

   - **VLAN ID**

     (Optional) Defines the VLAN ID of your network.

   - **Domain Name Server**

     Defines the name or IP address of the Domain Name Server (DNS) of your network.

     If you don't specify the DNS details, then the CA Access Gateway Administrative UI will not be protected automatically.

8. Click OK..

After provisioning the instance, you can access the instance on the virtual machine to configure the proxy rules and other configurations using the CA Access GatewayAdministrative UI.

Access the CA Access Gateway Administrative UI

By default, the installer creates a protection policy to protect the CA Access Gateway Administrative UI. The installer uses the defined Agent Name to create the protection policy with the following details:

- Domain: DOMAIN-SPSPADMINUI
- Policy: POLICY-SPSADMINUI

The protection policy does not contain the user directory information. Perform the following steps to log in to the CA Access Gateway Administrative UI:

1. Update DOMAIN-SPSPADMINUI with the user directory information.
2. Update POLICY-SPSADMINUI with user information.

Perform the above steps using the CA Single Sign-On Administrative UI.

You can launch the CA Access Gateway Administrative UI after you start the proxy engine services. To launch the URL, enter the following URL in a web browser:

http://*fully_qualified_hostname*:Tomcat_http_port/proxyui/

Tomcat_http_port: 8080

If you enable SSL on Tomcat, use the following link to access the CA Access Gateway Administrative UI.

https://*fully_qualified_hostname*:Tomcat_https_port/proxyui/

Tomcat_https_port: 8443

Post-deployment Management of the Virtual Machine

After provisioning the instance, access the virtual machine to configure CA Access Gateway. You can access the virtual machine through SSH.

Multiple instances on the same virtual machine are not supported.

We recommend that you do not install any other third-party software in the virtual machine that has the appliance installed.

Follow these steps:

1. SSH to the virtual machine using the following default credentials:
   Username: spsadmin
   Password: Proxy@inst
   You are prompted to change the password after initial login.

2. To start or stop the instance, you must be logged in as the root user. To access as the root user, type the following command:

   ```
   su root
   ```

3. Specify the following default password to access CA Access Gateway:
   Password: SPS@inst
   You are prompted to change the password after initial login.

4. Navigate to the following location to access CA Access Gateway to configure proxy rules and other configurations:

   ```
   \opt\CA\secure-proxy\
   ```

   For more information about configuring proxy rules, see the CA Access Gateway documentation in the Wiki.

Post-deployment Management of the Instance

Once the instance is provisioned, you can perform various actions such as Start, Shut Down, Reboot, on the virtual machine where the instance is running. You can also edit or delete an instance.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. In the left pane, click CA Access Gateway, Instances.
4. Select an instance that you want to manage.
5. Select one of the following options that you want to perform from the Action drop-down list:

- **Shut Down**
  Shuts down the selected instance.
  The color of Instance State changes to red to indicate that the instance is down.
- **Start**
  Starts the virtual machine and the instance that was previously shut down.
- **Reboot**
  Reboots the virtual machine and starts the instance.
- **Force Shut Down**
  Forcibly shuts down the selected instance. The color of Instance State changes to red to indicate that the instance is down.
- **Force Reboot**
  Forcibly reboots the virtual machine and starts the instance.
- **Console**
  Opens the console mode of the virtual machine in which the instance is running. Provide the login credentials of the virtual machine to access the instance.
- **Ping**
  Pings the virtual machine on which the instance is running to verify the availability of the virtual machine on the network and displays the status.
- **TraceRoute**
  Displays the details of data transfer between the NetScaler SDX UI and the instance.
- **Rediscover**
  Updates the version information of the instance after an upgrade.
  Prompts whether you want to run the Rediscover utility. Confirm to run the utility. Click the arrow next to the Instance name to view the details.

- **Upgrade**
  Upgrades the instance to the version that you want to upgrade to.

- **Upgrade SDX Tools**
  Upgrades the version of the SDX tools to which you want to upgrade to.

This section contains the following subsections:

Backup the Instance

You can back up the instance that is running on the virtual machine at any time, with all the required configurations. The backup file contains configuration information of all the instances that were available in the UI during the backup.

The backup operation takes a backup of the entire secure-proxy folder. You can use the backed-up file to restore the instance later.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. Click Management Service, Backup Files.
4. Click Back Up and click Yes.
   The details of backed up file are listed in the Backup Files page.

All the files except the SmHost.conf and SmHostFlow.conf that are available in the following folder are backed-up

/opt/CA/secure-proxy

Restore the Instance

You can restore an instance to its previous configuration at any time using the backup file that was taken earlier. In the NetScaler UI, a restore action deletes the existing instance, provisions a new instance, and copies the backed-up files.

After restoring the instance, use the default credentials that were used to log in to the virtual machine. The restore process creates a new virtual machine every time..

Follow these steps:

1. Log in to the NetScaler SDX UI.

2. Click the Configuration tab.
3. Click Management Service, Backup Files.
4. Select a backup file from which you want to restore the instance.
5. Click Restore.
6. Select the virtual machine on which you want to restore the instance.
7. Click OK.
   A confirmation message appears after the instance is restored on the virtual machine successfully.
8. Reboot the virtual machine to start the instance.

The restore action resets the credentials of the virtual machine. Use the default credentials to log in to the virtual machine.

Upgrade the Instance

Before you plan to upgrade an instance, verify that you have uploaded the latest version of the CA Access Gateway software to the NetScaler SDX UI.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. In the left pane, click CA Access Gateway, Instances.
4. Select an instance that you want to upgrade.
5. Select Upgrade from the Action drop-down list.
6. Select the version to which you want to upgrade from the Software Image drop-down list.
7. Click OK.
   The instance is upgraded to the selected version.

Reconfigure the Instance

Reconfigure the instance in one of the following scenarios:

- If you want to use a different CA Single Sign-On Policy Server other than the one that is currently configured.
- If the configuration parameters of the instance such as ACO or HCO change.

Before you attempt to reconfigure an instance, verify that the CA Single Sign-On policy administrator has deleted the existing trusted host objects.

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. In the left pane, click CA Access Gateway, Instances.
4. Select an instance that you want to reconfigure.
5. Click Edit.
6. Update the configuration parameters.
7. Click OK.
   The selected instance is reconfigured.

Generate a Technical Support File

If there are any issues with the instance, contact CA Support. Before you contact, generate a technical support file and send it to CA Support.

You can generate the technical support file using the NetScaler SDX UI. If you are unable to create the technical support file using the NetScaler UI for some reason, generate the file manually using the virtual machine.

Using the NetScaler SDX UI

Follow these steps:

1. Log in to the NetScaler SDX UI.
2. Click the Configuration tab.
3. Click Diagnostics, Technical Support.
4. Click Generate Technical Support File.
5. Select *one* of the following modes from the Mode drop-down:
   - Management Service (Including Instances)
     Contains the configuration files of the instance, the Management Service files, and SDX archive files.
   - Appliance (Including Instances)
     Contains the XenServer files, the configuration files of the instances, the Management Service files, and SDX archive files.

The Select Instances box appears.

6. Click Add.
7. Select an instance for which you want to generate the Technical Support file from the list in the Free Instances box.
8. Move the selected instance to the corresponding box.
9. Click OK.
   The file is created and listed in the Technical Support page.

Using the Virtual Machine

Follow these steps:

1. Log in to the virtual machine as a root user using the following credentials:
   Username: spsadmin
   Password: Proxy@inst

   If you had changed the default password earlier, then use the password that you have set.

2. Navigate to the following location:

   ```
   /var/opt/citrix/sdxtools/install/custom_scripts/lin/perl
   ```

3. Run the following command:

   ```
   perl op_tech_support.pl (http://op_tech_support.pl/)
   ```

4. The technical support file is created in the following location:

   ```
   /var/opt/citrix/sdxtools/install/software_files/tech_support
   ```

5. The technical support file has the following format:

   ```
   supportmaterials_<timestamp>.tar.gz
   ```

CA Access Gateway for NetScaler SDX Troubleshooting

The following sections describe some issues that you may encounter after deploying the instance.

Instance Fails to Start

Symptom:

After provisioning the instance, the instance fails to start automatically.

Solution:

See the vmbootsps_timestamp.log file available at the following location to know the possible causes:

`opt/CA/secure-proxy/proxy-engine/logs`

SSL Mode of the Instance

Symptom:

After provisioning the instance, unable to know the SSL mode of the instance.

Solution:

See the vmbootsps_timestamp.log file available at the following location to know whether the instance is running on SSL mode or not:

`opt/CA/secure-proxy/proxy-engine/logs`

Issues with Managing an Instance

Symptom:

Facing some issues while performing actions such as restore, upgrade, or backup on the instance.

Solution:

See the sdxtools.log file at the following location:

`/var/opt/citrix/sdxtools/install/log`

Actions that are taken to manage the instance are available in this log file.

Integrating CA SiteMinder® SPS with CA SiteMinder®

How CA Access Gateway Interacts with CA Single Sign-On

CA Access Gateway contains a Web Agent that is compatible with CA Single Sign-On Web Agent and Policy Server. CA Access Gateway must be configured as an object in CA Single Sign-On. Policies must be created that determine authentication and authorization requirements for accessing destination servers.

Configure the following objects using the CA Single Sign-On Administrative UI:

- An agent object with settings for Web Agent included in CA Access Gateway. Specify this Web Agent when creating realms.
- Connections to any user directories that authenticate and authorize users.
- Policy domains that contain realms, rules, and policies.
- Realms that contain resources you want to protect with CA Single Sign-On.
- Rules that identify specific resources and actions that you want to protect with CA Single Sign-On.
- Responses that can return information to applications, or to CA Access Gateway. Information returned to CA Access Gateway can determine how to route user requests.
- Policies that bind users and groups to rules and responses.

Authentication Scheme Considerations

CA Single Sign-On enforces authentication schemes to protect resources. When users attempt to access protected resources through a CA Single Sign-On Web Agent or CA Access Gateway, CA Single Sign-On asks for credentials based on the authentication scheme protecting the resource.

CA Single Sign-On also provides protection levels to each authentication scheme. The protection levels are enforced during single sign-on when the user tries to access resources protected by different authentication schemes. In such scenarios, the users can access resources protected by different authentication schemes without reauthentication if the protection levels for each of the authentication schemes are equal or lower. When moving from a lower protection level to a higher protection level, the user is challenged for authentication. When moving from a higher protection level to a lower protection level, the user is not challenged for reauthentication.

When CA Access Gateway is integrated with CA Single Sign-On, CA Access Gateway behaves similar to a CA Single Sign-On Web Agent. However, CA Access Gateway using any authentication scheme behaves similar to a Web Agent only if CA Access Gateway is configured to use default SessionCookieScheme scheme to track user sessions. If CA Access Gateway is configured to use any of the other advanced or cookieless session schemes, the user has to reauthenticate. Single sign-on does not work.

Proxy-Specific WebAgent.conf Settings

A number of settings in the WebAgent.conf configuration files for Web Agents installed behind the DMZ in an enterprise have specific implications for CA Access Gateway.

The settings that must be modified in the destination server WebAgent.conf files include:

proxytrust

Specifies how the destination server web agent interacts with CA Access Gateway. If the value is set to yes, the destination server web agent automatically trusts the authorizations made by CA Access Gateway. If the value is set to no, the destination server web agent requests authentication every time.

proxytimeout

Specifies the number of seconds the reverse proxy server waits for the agent that is deployed behind it to respond to a request.

Default: 120 seconds

Avoiding Policy Conflicts with Destination Server Web Agents

In some deployments, when CA Access Gateway is running in proxy trust mode, CA Access Gateway can protect resources from one set of users, while a Web Agent on a destination server protects the same resources from another set of users.

When creating policies, administrators must be sure that the policies do not conflict with each other. If policies contradict one another, it is possible that CA Single Sign-On may allow unwanted or unexpected behavior.

Configuring SiteMinder Rules that Redirect Users

CA Single Sign-On provides the ability to create response objects that redirect a request under certain conditions. For example, you can create a response that redirects a request to a custom error page after a failed authentication (OnRejectRedirect). By default for cookieless session schemes that rewrite a requested URL (Simple URL Rewriting), CA Access Gateway recognizes a user session information after a redirection.

To terminate a user session after a redirection, create a response attribute in CA Single Sign-On for the relevant policy that modifies the value of the CA Single Sign-On SM_REWRITE_URL header. This HTTP header must be set to NO to force a new session after a redirection.

For example, if you have a resource in realmA that is protected by an authentication scheme with a protection level of 5, and a second resource in realmB protected by authentication scheme with a protection level of 10, a user who successfully authenticates in realmA will be challenged for credentials when moving to realmB (due to the higher protection level).

If an OnRejectRedirect response is associated with realmB and the user fails to authenticate when challenged for credentials in realmB, the default behavior of CA Access Gateway maintains the user's original session information even after the user is redirected to a custom error page.

To terminate the user's session after the redirect and force an entirely fresh session on the next login attempt, you must create a response attribute that sets the SM_REWRITE_URL=NO, and associate the response with the appropriate policy.

CA Access Gateway and ERP Resources

You can use CA Access Gateway to secure resources managed by an ERP system. CA Access Gateway can function as a proxy in front of ERP agents protecting the following ERP systems:

- Siebel Application Server
- PeopleSoft Application Server
- SAP AS Web Application Server
- SAP ITS Application Server

While the ERP agent must be installed on the ERP server, CA Access Gateway secures the ERP resources at Policy Server.

To configure CA Access Gateway as a reverse proxy for an ERP agent, perform the following steps:

1. Specify the ERP server and appropriate port number for the <nete:forward> element in the proxyRules.xml file.
   For example:
   <!-- Proxy Rules–>
   <nete:proxyrules xmlns:nete="http://www.ca.com/">
   <nete:forward>http://server1.erpsrv.com:9080$0</nete:forward>
   </nete:proxyrules>

2. Specify the following values in the server.conf file:

   a. Set the value of the enableredirectrewrite parameter to "yes".

   b. Set the value of the redirectrewritablehostnames parameter to the host name of the system on which the ERP server is running.
      For example:
      <VirtualHost name="sales">
      hostnames="sales, sales.company.com"
      enableredirectrewrite="yes"
      redirectrewritablehostnames="server1.company.com,domain1.com"
      </VirtualHost>

3. Set the value of the addquotestocookie parameter in the <Sever> section of the server.conf to "no".

4. addquotestocookie="no"

Firewall Considerations

When configuring firewalls for the DMZ that contain CA Access Gateway, CA Access Gateway uses ports 8005 and 8009 for shutdown and AJP communication. These ports should be protected from access by entities outside of the DMZ.

You can change the ports used by CA Access Gateway by altering the appropriate directives in the server.conf file.

Keep Alive and Connection Pooling

CA Access Gateway is designed to use a connection pool to provide better performance by spreading out the workload generated from initiating server connections. It is recommended for performance reasons that KEEP ALIVE settings should be turned on for destination servers.

All destination server products have individual methods and attributes for managing keep alive settings. These settings should be reviewed and understood when configuring CA Access Gateway.

HTTP Header Configuration for Sun Java Web Servers

By default, some web servers, such as Sun Java Web servers limit the number of header variables that can accompany a request. You might have to increase this upper limit to accommodate transactions that contain many custom headers.

The server typically returns 413 Request Entity Too Large error is # of headers is greater than allowable maximum. For more information refer to your destination server's administration guide.

To change the maximum number of headers, perform the following steps:

1. Locate the magnus.conf file for the back-end Sun Java Web server and open it in a text editor.

2. Add or modify the following entry in magnus.conf :
   MaxRqHeaders 50

3. Set the maximum value at a level above the number of headers created by your CA Access Gateway transactions.

4. Restart the Sun Java Web server so that the changes will take affect.

HTTP Header for SiteMinder Processing with SPS

CA Access Gateway introduces an additional layer in the traditional CA Single Sign-On architecture. This layer forwards or redirects all requests to destination servers in the enterprise. When CA Access Gateway processes a request, the URL requested by the user is preserved in an HTTP header variable called SM_PROXYREQUEST. This header may be used by other applications that require the original URL requested by a user before CA Access Gateway proxied the request.

The value of the ProxyAgent parameter in the Agent Configuration object must be set to YES to enable sending the SM_PROXYREQUEST HTTP header to the backend.

Note: This header is only added when a request is made for a protected resource.

Handling Encoded URLs

Web servers can process both encoded and decoded normalized or unescaped URLs. How a Web server handles encoded URLs differs based on the type of server. For security reasons, and to provide consistent behavior, CA Access Gateway always decodes or normalizes a URI before processing. This provides a universal representation for a single URL, and protects against attempts to exploit CA Access Gateway using encoded strings.

Integrations

This section lists the products that can be integrated with CA Single Sign-On.

Integration with CA Arcot A-OK

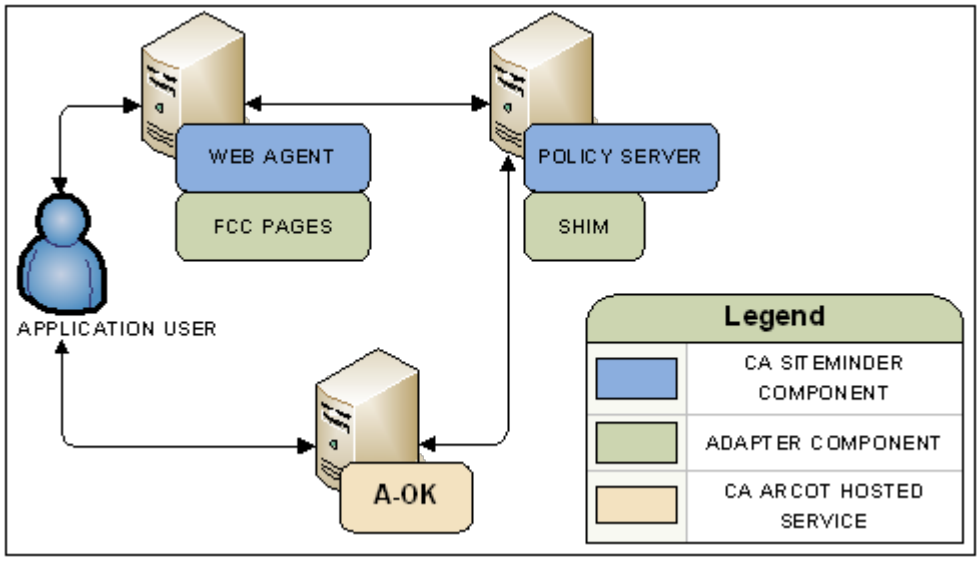
Contents

You use the CA Arcot A–OK Adapter™ (A–OK Adapter) to integrate CA Single Sign-On with the hosted CA Arcot A–OK service.

Note: The integration requires a minimum version of the A–OK Adapter. For more information about the supported version, see the CA Single Sign-On Platform Support Matrix.

The purpose of the following diagram is to:

- Illustrate how the A–OK Adapter and its components integrate in a CA Single Sign-On environment.
- Detail the major components and their general relationships.



SM--siteminder and arcot aok integration architecture

Authentication in a Hosted CA Arcot Integration

CA Arcot A–OK assumes authentication services in an integrated environment by guiding users through the authentication and risk evaluation processes. CA Arcot A–OK uses a series of SAML requests and responses to step through the authentication workflow.

The result of the risk evaluation is a risk score and corresponding advice, which is a recommend action, such as allow or deny the authentication.

CA Arcot A–OK forwards the advice to the Policy Server, which if necessary, continues with authorization services.

Confidence Levels and CA Single Sign-On Authorization

The Policy Server maintains authorization services in an integrated environment and can apply the risk score to authorization decisions. The risk score is created during the authentication process (see page 245).

The Policy Server applies the risk score as a CA Single Sign-On confidence level. A confidence level is based on a risk score, and as such, is also an integer that represents the likelihood that the transaction is safe.

You can apply a confidence level to both access management models:

- If you are protecting resources with policies, you can apply a confidence level to the following objects:
  - a policy realm
  - an active policy expression

- If you are protecting resources with EPM applications, you can apply a confidence level to the following objects:
  - an application component
  - an application role that comprises of a named expression that references the SM_USER_CONFIDENCE_LEVEL CA Single Sign-On generated attribute.

> **Note:** Applying a confidence level to a policy realm or an application component requires that you enable confidence level support (see page 248). Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.

The following example workflow details the relationship between both values and explains how the Policy Server applies a confidence level to authorization decisions:

1. After the user is successfully authenticated, the A–OK Adapter converts the risk score to a confidence level using the following algebraic formula:

   ```
   (100-risk score) * 10 = confidence level
   ```

2. The A–OK Adapter inserts the confidence level into the CA Single Sign-On session ticket.

3. The user requests protected resources, the Policy Server compares the confidence level in the session ticket to the confidence level configured in the policy or application.

4. The following actions can occur:
   - If the policy rule is configured to allow access and the confidence level of the user is equal to or greater than the confidence level configured in the policy realm or the active policy expression, the policy rule is triggered.

     > **Note:** If the confidence level of the user is less than the confidence level configured in the policy, CA Single Sign-On denies access.

   - If the policy rule is configured to reject access and the confidence level of the user is less than the value configured in the policy realm or the active policy expression, the policy rule is triggered.
   - If the confidence level of the user is less than the confidence level configured in the application role, the user is excluded from the role membership and CA Single Sign-On denies access.
   - If the confidence level of the user is equal to or greater than the confidence level configured in the application component, CA Single Sign-On grants access.

Risk Scores and Confidence Levels Compared

Although a risk score and a confidence level both ensure that the transaction is safe, there are differences between both values. Consider the following differences when planning for authorization decisions:

CA Arcot Risk Score	CA Single Sign-On Confidence Level
A numeric scale (0–100) represents a risk score.	A numeric scale (0–1000) represents a confidence level.
The lower the risk score, the greater the chance that the transaction is safe.	The higher the confidence level, the greater the chance that the transaction is safe. **Note:** A value of zero (0) represents no confidence. No confidence results in CA Single Sign-On denying access to the requested resource.

The following example workflow details the inverse relationship between a risk score and a confidence level:

1. A user requests a CA Single Sign-On protected resource and is forwarded to CA Arcot A–OK for authentication.
2. The A–OK Adapter guides the user through authentication and risk analysis. Based on the CA Arcot A–OK evaluation and scoring rules, the user is authenticated with a risk score of 30. The lower risk score is representative of a safe transaction.
3. The A–OK Adapter:
    a. Forwards the authentication decision to the Policy Server.
    b. Converts the risk score to a confidence level using the following algebraic formula:

    ```
    (100 - risk score) * 10 = confidence level
    ```

    In this example, the A–OK Adapter converts the risk score to a confidence level using the following algebraic formula:

    ```
    (100 - 30) * 10 = 700
    ```

    The higher confidence level is representative of a safe transaction.
4. The A–OK Adapter inserts the confidence level into the session ticket of the user.
5. The user requests a resource that is protected by a policy or an application that requires a confidence level of at least 700.
6. The Policy Server grants access to the resource.

Enable Confidence Level Support

You can optionally apply a confidence level to authorization decisions. Consider the following items:

- You can apply a confidence level to the following objects:
  - A policy realm
  - An active policy expression
  - An application component
  - An application role that includes a named expression, which references the SM_USER_CONFIDENCE_LEVEL generated attribute.
- Enable confidence level support only to apply a confidence level to a realm or an application component. Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.

Follow these steps:

1. Log in to a Policy Server host system.
2. Start the XPSConfig utility.
   XPSConfig prompts for an option.
3. Enter **SM** and press Enter.
   The list of options displays and XPSConfig promtps for an option.
4. Filter the list by entering **F** and typing the string **Confidence.**
   The ConfidenceLevelSupportEnabled parameter displays.
5. Enter the number associated with the ConfidenceLevelSupportEnabled parameter to modify it.
6. Enter C to change the parameter value. The default is False.
   The pending value of the parameter changes to True.
7. Quit the XPSConfig utility.
8. Restart the Policy Server.

The confidence level support is now enabled.

CA Arcot A-OK Integration Use Cases

The following use cases detail how you can integrate CA Single Sign-On with CA Arcot A–OK strong authentication and risk evaluation. The use cases begin with a simple integration and progress into more complex scenarios.

CA Arcot A–OK Authentication and Risk Analysis

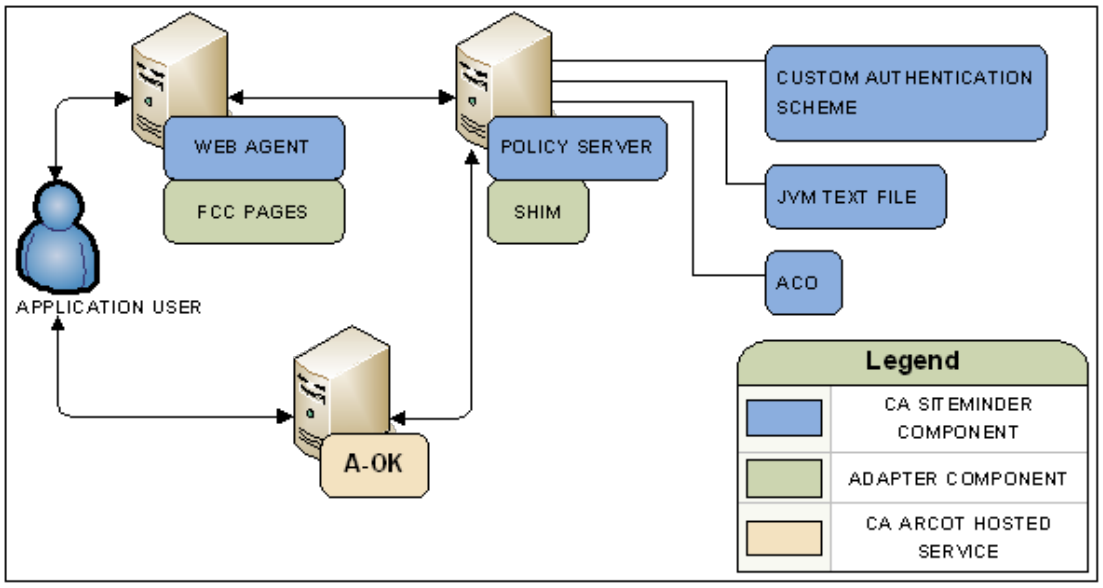
The simplest deployment includes integrating the A–OK Adapter and all related components with CA Single Sign-On.

The A–OK Adapter guides users through the authentication and risk evaluation processes to apply a risk score during the authentication process (see page 245).

Follow these steps:

1. Be sure that the CA Arcot A–OK service is available.
2. Install and deploy the A–OK Adapter and all related components. These components include a set of Forms Credential Collector files. These files let you use the A–OK Adapter HTML forms authentication scheme to gather user credentials.
3. Complete the following steps:
    a. Configure a CA Single Sign-On Custom authentication scheme to call the A–OK Adapter library.
    b. Determine which Web Agents are included in the CA Arcot A–OK integration. Configure the respective Agent Configuration Objects (ACO) to support the integration.
    c. Add the A–OK Adapter JAR files, certificates, and properties files to the Java Virtual Machine (JVM) file (JVMOptions.txt) of the Policy Server.

The following diagram illustrates this deployment scenario:



SM--aok auth and risk analysis

CA Single Sign-On Authorization and Confidence Levels

You can extend the Policy Server authorization services by adding a confidence level (see page) to both access management models.

Adding a confidence level lets you apply the CA Arcot A–OK risk analysis results to authorization decisions.

Follow these steps:

1. Complete the steps in CA Arcot A-OK Authentication and Risk Analysis (see page 249).
2. (Optional) If you plan on applying a confidence level to a policy realm or an application component, enable confidence level support (see page 248). Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.
3. Complete one of the following steps:
   - If you are using policies to protect resources, add a confidence level to one or more policy realms or active policy expressions.
   - If you are using applications to protect resources, add a confidence level to one or more application components or application roles.

User Store Considerations

All CA Single Sign-On users to which the integration applies must be made available to the CA Arcot A–OK hosted service.

Contact CA Arcot Support for assistance.

Integration with CA Application Performance Management

See the CA Application Performance Management documentation (https://docops.ca.com/ca-apm/10/en) for detailed procedures to integrate CA Single Sign-On with CA CEM and Application Performance Management.

Integration with CA DataMinder Content Classification Service

Contents

A CA Single Sign-On integration with the CA DataMinder Content Classification Service (CCS) lets the Policy Server use CCS content assessments to make content–aware authorization decisions.

Consider the following items before you begin:

- The integration requires a minimum version of CA Single Sign-On, the CCS, and the CA Single Sign-On Agent for SharePoint.

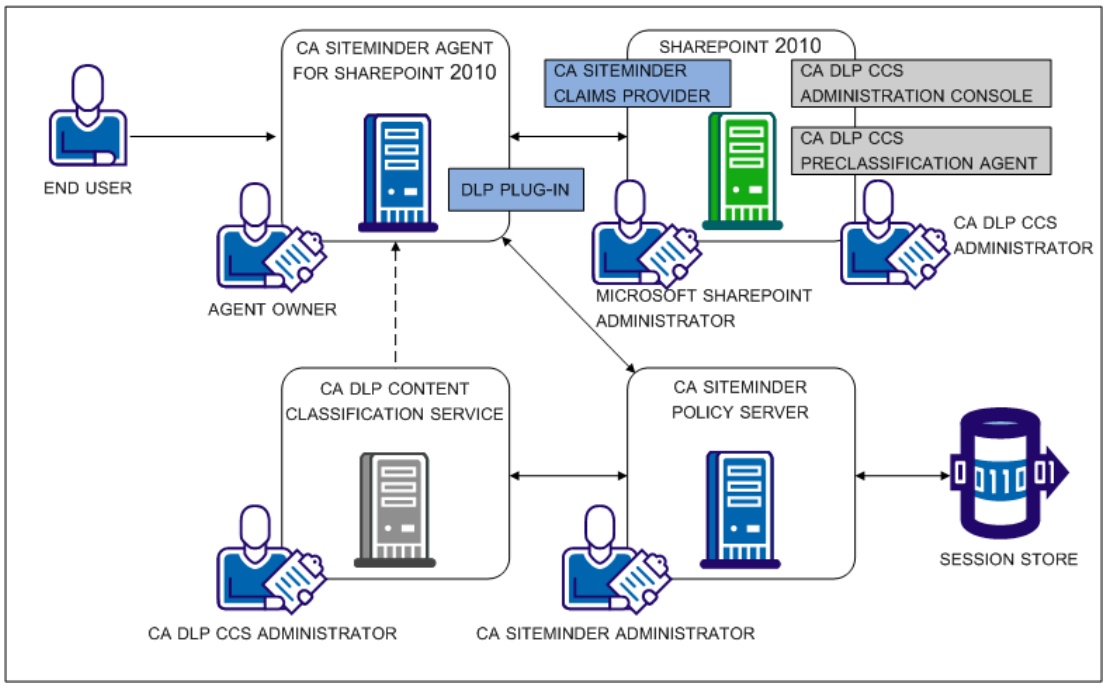
  

  Note: For more information, see the CA Single Sign-On Platform Support Matrix.

- Multiple organizational roles are required to enable the integration. Coordinate the integration with the following people:
  - A CCS administrator
  - A CA Single Sign-On administrator
  - The owner of the CA Single Sign-On agent for SharePoint

The purpose of the following diagram is to:

- Illustrate the general relationship between the CCS and CA Single Sign-On components in an integrated environment. The diagram is not intended to represent workflow or represent every component that is deployed in the integrated environment.
- Associate the individuals responsible for installing or configuring a required component.



SM--DLP Components Overview

CA DataMinder Content Classification Service

The role of the CCS in the integration is to make available predefined content classifications to the Policy Server. The classifications correspond to document types commonly found in a corporate environment. The Policy Server uses the classifications to make content–aware authorization decisions.

As the dotted line in CA DataMinder Content Classification Service (see page 250) illustrates, if a content classification is unavailable at the time of the Policy Server authorization decision, the CCS can request the resource directly to classify or re–classify it. The CCS:

- Passes the result to the Policy Server to make the authorization decision.
- Adds the result to the CCS classification cache for future authorization decisions.

CA DataMinder Content Classification Service Preclassification Agent

The role of the CA DataMinder CCS preclassification agent in the integration is to scan and classify SharePoint documents offline. Classifying documents offline avoids the need to retrieve a document classification as part of the Policy Server authorization decision.

CA Single Sign-On Policy Server

The role of the Policy Server in the integration is to act as the Policy Decision Point (PDP). The Policy Server:

- Maintains all authentication and authorization services in the integrated environment.
- Communicates with the CA Single Sign-On agent for SharePoint to retrieve the resource information of the protected document.
- Communicates with the CA DataMinder CCS to retrieve the content classification of the protected document. The Policy Server uses the results to make a content–aware authorization decision.

If configured to do so, the Policy Server can create a single use security token for the CA DataMinder CCS. The CA DataMinder CCS uses the token to request the resource directly. The CCS requests a resource when it must classify or re–classify it as part of the authorization decision.

CA Single Sign-On Agent for SharePoint

The role of the Agent for SharePoint in the integration is to act as the Policy Enforcement Point (PEP). The agent for SharePoint:

- Intercepts the request for the SharePoint document.
- Extracts the resource information of the document.
- Passes the resource information to the Policy Server.

CA Single Sign-On Session Store

The role of the session store is to make available single use security tokens to all Policy Servers in a clustered environment. If configured to do so, a Policy Server creates a security token for the CA DataMinder CCS. The token serves as credentials for the CA DataMinder CCS when it requires access to the protected document.

The CA DataMinder CCS requires access to a protected document when it cannot provide the content classification to the Policy Server. Requesting the resource lets the CCS:

- Classify or re–classify the document.
- Provide the content classification to the Policy Server.
- Add the content classification to the CA DataMinder classification cache for future Policy Server authorization requests.

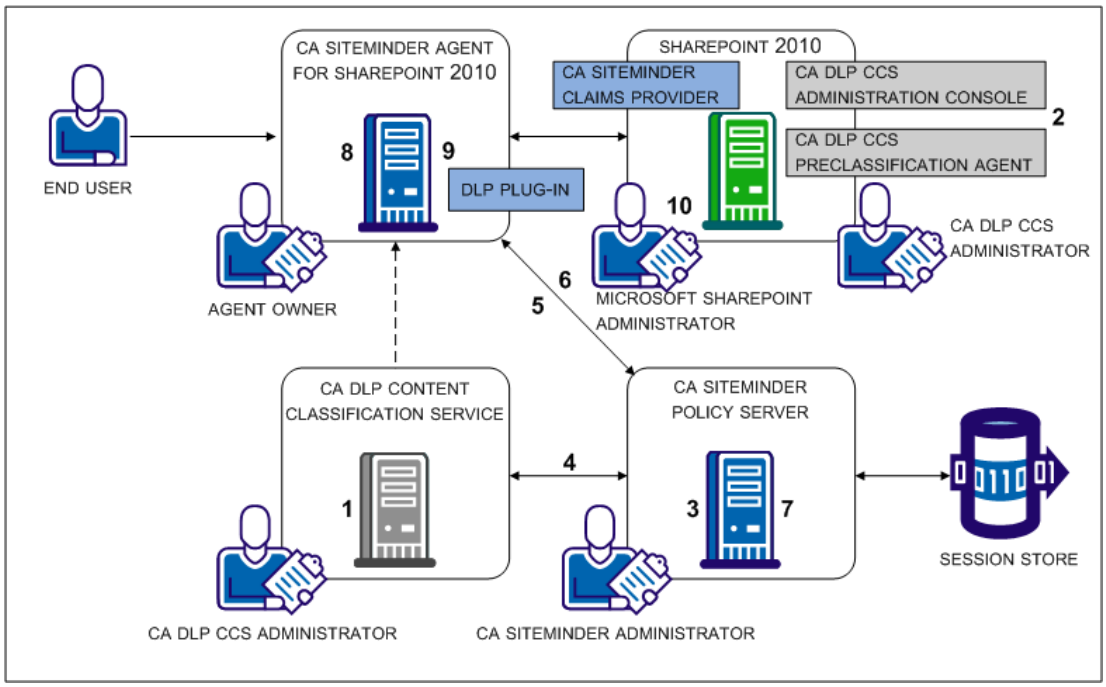
As part of the process, the agent for SharePoint returns the token to a Policy Server to validate authenticity. If the agent for SharePoint sends the validation request to a Policy Server that did not create the token and the environment:

- Includes a session store, the Policy Server retrieves the token, validates it, and authorizes the CA DataMinder CCS.
- Does not include a session store, the Policy Server cannot validate the token and denies the authorization request.

CA DataMinder Content Classification Service Integration Roadmap

The following diagram:

- Illustrates a sample CA DataMinder and CA Single Sign-On integration.
- Lists the order in which each component is installed and configured.



SM--CA DLP Integration Road Map

The following table includes each step in the figure and lists the individual responsible for the task.

Step	Action	Responsibility
1	Install and configure the CA DataMinder CCS to communicate over SSL (see page).	CA DataMinder CCS administrator
2	Install and configure the CA DataMinder preclassification agent (see page).	CA DataMinder CCS administrator

Step	Action	Responsibility
3	Enable SSL for the integration (see page 256).	CA Single Sign-On administrator
4	Configure a connection to the CA DataMinder CCS (see page).	CA Single Sign-On administrator
5	Modify the agent for SharePoint agent configuration object (see page 257).	CA Single Sign-On administrator
6	Enable the DLP exclusion list parameter (see page 258).	CA Single Sign-On administrator
7	Enable an authorization failure message (see page 258).	CA Single Sign-On administrator
8	Modify the proxy rules for SharePoint multi-authentication (see page 260).	SharePoint agent owner
9	Enable the DLP plug-in (see page 262).	SharePoint agent owner
10	Provide the CA DataMinder CCS with read access to SharePoint applications (see page 263).	SharePoint administrator

CA DataMinder CCS Administrator Tasks

The CA DataMinder CCS administrator is responsible for:

- Installing one or more CA DataMinder Content Classification Services and configuring each instance to communicate over SSL. The integration requires that the CA DataMinder Content Classification Service and the Policy Server to communicate securely.
  A CA Single Sign-On administrator requires the CCS server certificate file to enable SSL on Policy Server host systems.

  Important! Use the same certificate and password for all CCS instances when configuring them to communicate securely.

- Installing a CA DataMinder CCS preclassification agent to the SharePoint environment and scheduling classification service scans. The CA DataMinder CCS Administration console is installed with the preclassification agent.

CA Single Sign-On Administrator Tasks

The CA Single Sign-On administrator is responsible for enabling the environment for the integration. Complete the integration steps in the following order:

1. Enable SSL for the integration.
2. Configure the connection to the CA DataMinder CCS.
3. Modify the SharePoint agent configuration object.
4. Enable the DLP exclusion list parameter.

5. Enable an authorization failure message.

Enable SSL for the Integration

The integration requires that the CA DataMinder CCS and the Policy Server communicate securely.

- A CA DataMinder CCS administrator is required to configure all CA DataMinder CCS instances to communicate securely. Request the CCS server certificate from the CA DataMinder administrator before you begin. The server certificate is required to enable SSL for the integration.
- Enabling SSL is a local setting. Complete the following procedure for each Policy Server that is protecting SharePoint documents.

Follow these steps:

1. Create a client certificate chain file. A chain file is a single file that contains the certificate file and the respective private key.

   Important! The file must be in PEM format.

2. Log in to the Policy Server host system.
3. Deploy the CCS server certificate and client certificate chain file.
4. Navigate to *siteminder_home*\bin\thirdparty\axis2c.
5. Open the following file:

   `axis2.xml`

6. Locate the SERVER_CERT parameter. Replace the sample value with the path to the CCS server certificate file.
7. Locate the KEY_FILE parameter. Replace the sample value with the path to the client certificate chain file.
8. Locate the SSL_PASSPHRASE parameter. Replace the sample value with the passphrase that is used to encrypt the private key in the client certificate chain file.
9. Save the file.

Configure a Connection to a CA DataMinder Content Classification Service

The Policy Server requires a connection to a CA DataMinder CCS to:

- Retrieve the content classification of a protected document.
- User the content classification to make a content–aware authorization decision.

Configuring the connection is a local setting. Complete the following procedure for every Policy Server that is protecting the SharePoint documents.

Follow these steps:

1. Log in to the Administrative UI with a superuser administrator account.
2. Click Policies, Configure DLP.
3. Select True from the CA Single Sign-On DLP Integration Enabled list.
4. Enter the IP address or fully qualified domain name of the primary CA DataMinder CCS.
5. (Optional) Enter additional configuration parameters.

   Note: For more information about the parameters, click Help.

6. Click Save.
7. Restart the Policy Server to enable the Policy Server for the integration and to configure the connection to the CA DataMinder CCS.
8. Restart any Administrative UI that is registered with the Policy Server that has been restarted.

Modify the SharePoint Agent Configuration Object

Modifying the SharePoint agent configuration object configures the agent to extract resource information from the protected document. The agent passes the information to the Policy Server as part of the authorization process.

Follow these steps:

1. Log in to the Administrative UI.
2. Click Infrastructure, Agent Configuration Objects.
3. Locate the agent configuration object for your SharePoint 2010 agents.
4. Click the edit icon to open the object.
5. Enter the following value for the DLPSupportEnabled parameter:

   ```
   SHAREPOINT
   ```

6. Click Submit.
   The agent configuration object is enabled for the integration.

7. Contact the agent for SharePoint owner. The agent configuration object is the Policy Server counterpart to the web agent configuration file. A separate procedure is required on the web tier to complete the integration for the agent for SharePoint. The agent for SharePoint owner is responsible for completing the task.

Enable the DLP Exclusion List Parameter

The SharePoint 2010 agent configuration object includes the DLP exclusion list parameter. This parameter contains a set of default resources that the Policy Server excludes from CA DataMinder CCS content classifications. Excluding resources from content classifications indicates to SharePoint agents that the resource can be automatically authorized.

The integration requires that you enable the parameter.

Follow these steps:

1. Log in to the Administrative UI.
2. Click Infrastructure, Agent Configuration Objects.
3. Locate the agent configuration object for your SharePoint 2010 agents.
4. Click the edit icon to open the object.
5. Locate the following parameter:

   ```
   #DlpExclusionList
   ```

6. Click the edit icon to open the parameter.
7. Remove the pound sign from the parameter name.
8. If you want to exclude additional resources from content classifications, add the extension to the default set.

   Note: Separate the values with a comma.

9. Click OK.
10. Click Submit.
    The agent configuration object is enabled.

Enable an Authorization Failure Message

By default, when users fail a DLP content check during authorization, they are redirected to a standard HTTP 403 error message.

Enable authorization failure messages to return an alternate, user–friendly message.

Follow these steps:

1. Create the custom error page using either a text file or an HTML file. Consider the following items:
   - You can only redirect users to a custom error page. Applications are not supported
   - If your environment uses Internet Explorer and you are deploying a custom HTML file, include:
     - A style element in the head element
     - A trailing line before you close the body element

     The HTML file requires these items to prevent Internet Explorer from displaying the standard error message, instead of your custom page.
2. Log in to the Administrative UI.
3. Click Infrastructure, Agent Configuration Objects.
4. Locate the agent configuration object for your SharePoint 2010 agents.
5. Click the edit icon to open the object.
6. Locate the following parameter:

   ```
   #DlpErrorFile
   ```
7. Click the edit icon to open the parameter.
8. Remove the pound sign from the parameter name.
9. Enter the location of the custom error page in the Value field.
   Example:

   ```
   C:\custompages\dlperror.txt
   ```
10. Click OK.
11. Click Submit.
    The user–friendly message is enabled.

CA Agent for SharePoint Owner Tasks

The CA Agent for SharePoint administrator is responsible for enabling the SharePoint agent environment for the integration. Complete the integration steps in the following order:

1. If SharePoint is configured for multi–authentication mode, modify the proxy rules.
2. Enable the DLP plug–in.

Modify the Proxy Rules for SharePoint Multi–Authentication

If SharePoint is configured for multi–authentication, specific Agent for SharePoint proxy rules is required to ensure that the CA DataMinder CCS classifies your SharePoint resources properly.

Contact the Sharepoint administrator to determine if multi–authentication is configured. If multi–authentication is configured, complete the following procedure.

Important! Do not use any other proxy rule settings when the SharePoint environment is configured for multi–authentication. The CA DataMinder CCS request for resources uses an HTTP header for proper forwarding by the Agent for SharePoint. If the Agent for SharePoint does not properly forward these requests using the following proxy rules, unauthorized access and disclosure of your protected information is possible.

Follow these steps:

1. Locate the following file on your Agent for SharePoint:

   ```
   Agent-for-SharePoint_home\proxy-engine\conf\proxyrules.xml
   ```

2. Rename the previous file using a name similar to the following example:

   ```
   proxyrules_xml_default.txt
   ```

3. Open the following file on your CA Single Sign-on Agent for SharePoint with a text editor:

   ```
   Agent-for-SharePoint_home\proxy-engine\examples\proxyrules\proxyrules_example2.
   xml
   ```

4. Save the previous file as a new file in the following location:

   ```
   Agent-for-SharePoint_home\proxy-engine\conf\proxyrules.xml
   ```

5. Locate the following text in the updated proxyrules.xml file:

   ```
   :///$$PROXY_RULES_DTD$$"
   ```

6. Replace the previous text with the following text:

   ```
   :///C:\Program Files\CA\Agent-for-SharePoint\proxy-engine\conf\dtd\proxyrules.
   dtd"
   ```

7. Locate the following text:

   ```
   http://www.company.com
   ```

8. Change the previous text to the domain of your organization. Use the following example as a guide:

```
http:www.example.com
```

9. Locate the following line:

```
<nete:cond type="header" criteria="equals" headername="HEADER">
```

10. Edit the previous line to match the following line:

```
<nete:cond type="header" headername="SMSERVICETOKEN">
```

11. Locate the following line:

```
<nete:case value="value1">
```

12. Edit the previous line to match the following line:

```
<nete:case value="DLP">
```

13. Add a line after the previous line.

14. Copy and paste the following xml syntax onto the new line:

```
<nete:xprcond>
<nete:xpr>
<nete:rule>^/_login/default.aspx\?ReturnUrl=(.*)</nete:rule>
<nete:result>http://sharepoint.example.com:port_number/_trust/default.aspx?
trust=siteminder_trusted_identity_provider&amp;ReturnUrl=$1</nete:result>
</nete:xpr>
<nete:xpr-default>
<nete:forward>http://sharepoint.example:port_number$0</nete:forward>
</nete:xpr-default>
</nete:xprcond>
```

15. Replace both instances of the **sharepoint.example:***port_number* in the previous section with *one* of the following values:

    - The host name, domain, and port number of your hardware load balancer. This hardware load balancer operates between your CA Single Sign-on Agent for SharePoint server and the SharePoint servers.

    - host name, domain, and port number of your single web front end. In this context, this web front end (WFE) refers a web server that operates in front of your "back end" SharePoint servers.

16. Replace the instance of *siteminder_trusted_identity_provider* in the previous section with the name of your CA Single Sign-On trusted identity provider.

17. Locate the following line in the file:

```
<nete:forward>http://home.company.com</nete:forward>
```

18. Replace the **home.company.com** in the previous line with *one* of the following values:

- The host name, domain, and port number of your hardware load balancer. This hardware load balancer operates between your CA Single Sign-on Agent for SharePoint server and the SharePoint servers.

- host name, domain, and port number of your single web front end. In this context, this web front end (WFE) refers a web server that operates in front of your "back end" SharePoint servers.

19. Save the file and close your text editor.
    The proxy rules are set.

Enable the DLP Plug–in

Enabling the DLP plug–in configures the agent to extract the resource information from the protected document. The agent passes the information to the Policy Server as part of the authorization process.

Important! A separate procedure is required in the application tier to enable the integration. Do not modify the web agent configuration file before the SharePoint agent configuration object is modified. The CA Single Sign-On administrator is responsible for completing the task.

Follow these steps:

1. Log in to the system hosting your Agent for SharePoint.

2. Go to the following location:

   ```
   Agent-for-SharePoint_Home\proxy-engine\conf\defaultagent
   ```

   - **Agent-for-SharePoint_Home**

     Indicates the directory where the CA CA Single Sign-on Agent for SharePoint is installed.

     Default: (Windows) [32-bit] C:\Program Files\CA\Agent-for-SharePoint

     Default: (Windows) [64-bit] C:\CA\Agent-for-SharePoint**Default**: (UNIX/Linux) /opt/CA /Agent-for-SharePoint

3. Open the following file:

   ```
   WebAgent.conf
   ```

4. Uncomment (remove the # sign to the left of) the line that loads the disambiguation plug–in.
   Example: (Windows [32-bit]) LoadPlugin="C:\Program Files\CA\Agent-for-SharePoint\agentframework\bin\DisambiguatePlugin.dll"
   Example: (Windows [64-bit]) LoadPlugin="C:\CA\Agent-for-SharePoint\agentframework\bin\DisambiguatePlugin.dll"
   Example: (UNIX/Linux) LoadPlugin="/opt/CA/Agent-for-SharePoint/agentframework/bin /DisambiguatePlugin.so"

5. Save the file.

6. Restart the web server.
   The CA Single Sign-on Agent for SharePoint is configured for the CA DataMinder integration.

Microsoft SharePoint Administrator Task

The SharePoint Administrator is responsible for providing the CA DataMinder CCS with read access to the SharePoint applications that CA Single Sign-On is protecting. The CA DataMinder CCS requires read access to determine the types of content that protected documents contain.

Providing read access to the CA DataMinder CCS is local to each application. Complete the following procedure for every application that CA Single Sign-On is protecting.

Follow these steps:

1. If the CA Single Sign-On Claims provider is configured, the SharePoint loopback search feature is required. If the feature is not enabled, follow these steps:

   a. Click Start, All Programs, Microsoft SharePoint 2010 Products, SharePoint 2010 Management Shell.

   b. Use the management shell to go to the following directory:

      ```
      C:\Program Files\CA\SharePointClaimsProvider\scripts
      ```

   c. Enter the following command:

      ```
      .\Set-SMClaimProviderConfiguration.ps1 -EnableLoopBackSearch
      ```

   d. Loopback search is enabled.

2. Log in to SharePoint Central Administration.

3. Locate the Application Management section and click Manage web applications.
   A list of applications appears.

4. Select an application and click User Policy in the Web Applications ribbon.
   The Policy for Web Application dialog appears.

5. Click Add Users.
   The Add Users wizard appears.

6. Select a Time Zone and click Next.

7. Locate the Users field and click the browse icon.
   The Select People and Groups – Web Page dialog appears.

8. Locate the CA Single Sign-On trusted identity provider. Under the trusted identity provider, click the associated identifier claim.

9. Enter the following value in the Find field and click the search icon:

```
caservice
```

10. Double–click the following user icon and click OK.

    ```
    caservice
    ```

    The Add Users dialog appears.

11. Select the following permission and click Finish:

    ```
    Full Read - Has full read-only access.
    ```

    The Policy for Web Application dialog appears.

12. Click OK.
    The CA DataMinder CCS has read access to the application.

Integration with CA Identity Manager

To integrate CA Single Sign-On with CA Identity Manager, select the check box *CA Identity Manager Integration* in the Policy Server Configuration Wizard.

See the CA Identity Manager documentation (https://wiki.ca.com/pages/viewpage.action?pageId=216282531) for detailed procedures to integrate CA Single Sigh-On with CA Identity Manager.

Note: The video references the product name as CA SiteMinder, the former name of CA Single Sign-On (CA SSO).

CA Identity Manager Roles and Access Control

Integrating with CA Identity Manager lets you can implement policy–based access control using CA Identity Manager roles. These roles enable centralized management of user privileges in external applications.

For more information about configuring the integration, see the CA Identity Manager documentation.

The integration requires:

- The Policy Server installation includes the required data definitions for the CA Identity Manager integration at the following location.

  siteminder_home\xps\dd

siteminder_home specifies the Policy Server installation path.

- The name of the file is:

  ```
  IdmSmObjects.xdd
  ```

Important! Do not import this file in to the policy store until you have completed the CA Identity Manager integration. If you import the data definitions before completing the integration, the Policy Server can reach an indeterminate state. Coordinate the integration with your CA Identity Manager administrator.

- CA Identity Manager administrators manage environments and roles in CA Identity Manager to determine user access to the applications that CA Single Sign-On secures. The CA Single Sign-On Administrative UI refers to these environments as an IDM environment.

  For more information about environments and roles, see the CA Identity Manager documentation.

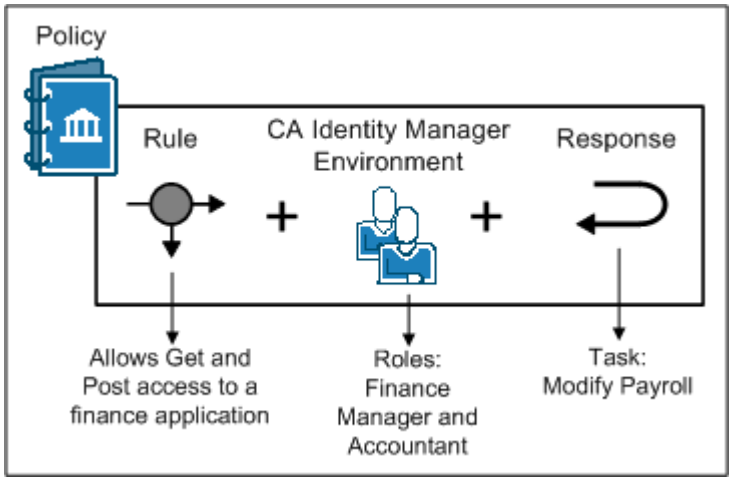
- Administrators use the Administrative UI to associate one or more IDM environments to a policy domain and user directory. An administrator cannot create or manage an IDM environment.

- Administrators use the Administrative UI to create a policy and associate one or more roles that are available to the IDM environments. An administrator cannot create or manage a CA Identity Manager role.

Note: You cannot apply a CA Identity Manager role to an enterprise management application.

CA Single Sign-On can also provide details about entitlements that a CA Identity Manager user has in protected applications. As the following figure illustrates, a CA Single Sign-On administrator associates a response with an access rule in the policy. The response contains a response attribute that specifies a CA Single Sign-On–generated user attribute.

The CA Single Sign-On–generated user attribute retrieves task information from CA Identity Manager. The Policy Server passes this information to the web agent as an HTTP header variable or a cookie. The web agent makes the header variable or cookie available to the protected application for fine–grained access control.



SM--Identity Manager Access Roles and Entitlements

Integration with CA Arcot WebFort and RiskFort

Contents

You can use the CA Arcot Adapter™ (Adapter) to integrate CA Single Sign-On with an on–premise implementation of the CA Arcot WebFort strong authentication solution and the CA Arcot RiskFort adaptive authentication solution.

Consider the following requirements before you begin:

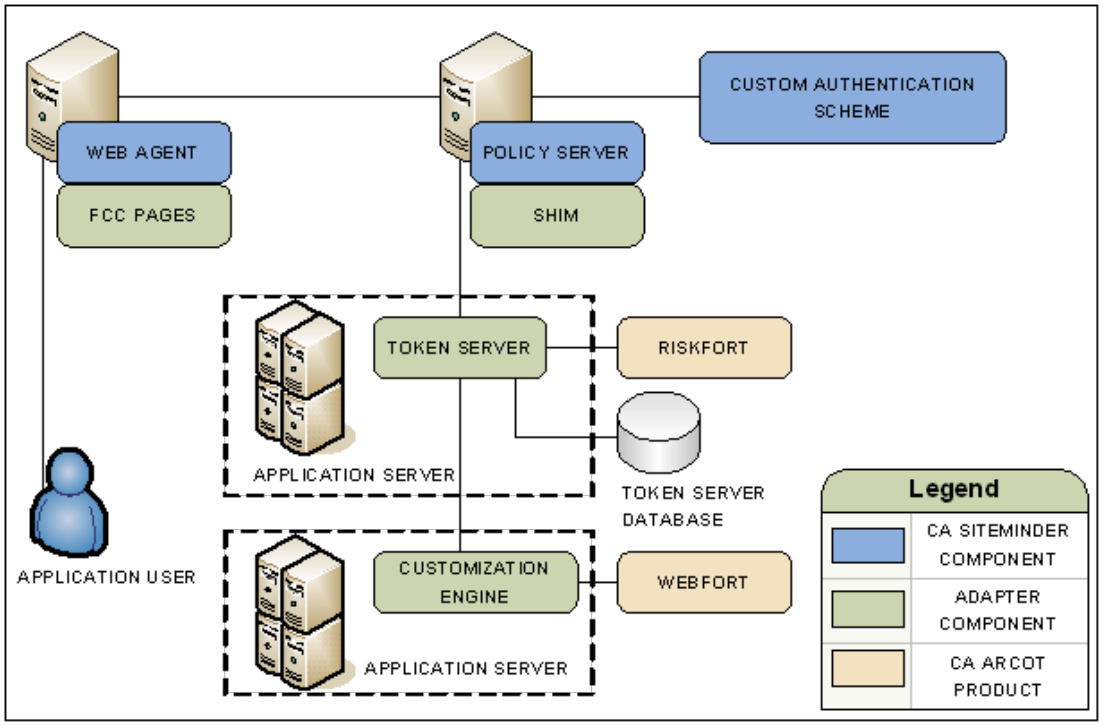
- The integration requires a minimum version of the Adapter and CA Arcot RiskFort.
- The integration requires a minimum version of CA Arcot WebFort.

Note: For more information about the supported versions, see the CA Single Sign-On Platform Support Matrix.

The purpose of the following diagram is to:

- Illustrate how the Adapter and its components, CA Arcot RiskFort, and CA Arcot WebFort integrate in a CA Single Sign-On environment
- Detail the major components and their general relationships

Note: For more information about installing and configuring all CA Arcot components, see the CA Arcot documentation.



SM--siteminder and arcot integration architecture

Authentication in an On–Premise Arcot Integration

CA Arcot assumes authentication services in an integrated environment by guiding users through the authentication (CA Arcot WebFort) and risk evaluation (CA Arcot RiskFort) processes. During the authentication process:

- CA Arcot WebFort provides the strong authentication, which helps to ensure that the identities of the users requesting the CA Single Sign-On–protected resources are legitimate.
- CA Arcot RiskFort collects a range of data to complete a risk evaluation, which determines the level of risk that is associated with each transaction.

The result of the risk evaluation is a risk score and corresponding advice, which is a recommend action, such as allow or deny the authentication.

CA Arcot forwards the advice to the Policy Server, which if necessary, continues with its authorization services.

Confidence Levels and CA Single Sign-On Authorization

The Policy Server maintains authorization services in an integrated environment and can apply the risk score to authorization decisions. The risk score is created during the authentication process (see page 267).

The Policy Server applies the risk score as a CA Single Sign-On confidence level (confidence level). A confidence level is based on a risk score, and as such, is also an integer that represents the likelihood that the transaction is safe.

You can apply a confidence level to both access management models:

- If you are protecting resources with policies, you can apply a confidence level to the following objects:
  - a policy realm
  - an active policy expression
- If you are protecting resources with EPM applications, you can apply a confidence level to the following objects:
  - an application component
  - an application role that is composed of a named expression that references the SM_USER_CONFIDENCE_LEVEL CA Single Sign-On generated attribute.

Note: Applying a confidence level to a policy realm or an application component requires that you enable confidence level support (see page 270). Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.

The following example workflow details the relationship between both values and explains how the Policy Server applies a confidence level to authorization decisions:

1. After the user is successfully authenticated, the Adapter converts the risk score to a confidence level using the following algebraic formula:

   ```
   (100-risk score) * 10 = confidence level
   ```

2. The Adapter inserts the confidence level into the CA Single Sign-On session ticket.
3. As the user requests protected resources, the Policy Server compares the confidence level in the session ticket to the confidence level configured in the policy or application.
4. The following actions can occur:
   - If the policy rule is configured to allow access and the confidence level of the user is equal to or greater than the confidence level configured in the policy realm or the active policy expression, the policy rule is triggered.

     Note: If the confidence level of the user is less than the confidence level configured in the policy, CA Single Sign-On denies access.

- If the policy rule is configured to reject access and the confidence level of the user is less than the value configured in the policy realm or the active policy expression, the policy rule is triggered.
- If the confidence level of the user is less than the confidence level configured in the application role, the user is excluded from the role membership and CA Single Sign-On denies access.
- If the confidence level of the user is equal to or greater than the confidence level configured in the application component, CA Single Sign-On grants access.

Risk Scores and Confidence Levels Compared

Although a risk score and a confidence level both ensure that the transaction is safe, there are differences between both values. Consider the following differences when planning for authorization decisions:

CA Arcot Risk Score	CA Single Sign-On Confidence Level
A numeric scale (0–100) represents a risk score.	A numeric scale (0–1000) represents a confidence level.
The lower the risk score, the greater the chance that the transaction is safe.	The higher the confidence level, the greater the chance that the transaction is safe. **Note:** A value of zero (0) represents no confidence. No confidence results in CA Single Sign-On denying access to the requested resource.

The following example workflow details the inverse relationship between a risk score and a confidence level:

1. A user requests a CA Single Sign-On protected resource and is forwarded to CA Arcot for authentication.
2. The Adapter guides the user through authentication and risk analysis. Based on the CA Arcot evaluation and scoring rules, the user is authenticated with a risk score of 30. The lower risk score is representative of a safe transaction.
3. The Adapter:
    a. Forwards the authentication decision to the Policy Server
    b. Converts the risk score to a confidence level using the following algebraic formula:

    ```
    (100 - risk score) * 10 = confidence level
    ```

    In this example, the Adapter converts the risk score to a confidence level using the following algebraic formula:

    ```
    (100 - 30) * 10 = 700
    ```

The higher confidence level is representative of a safe transaction.

4. The Adapter inserts the confidence level into the session ticket of the user.
5. The user requests a resource that is protected by a policy or an application that requires a confidence level of at least 700.
6. The Policy Server grants access to the resource.

Enable Confidence Level Support for Authorization Decisions

You can optionally apply a confidence level to authorization decisions. Consider the following items:

- You can apply a confidence level to the following objects:
  - A policy realm
  - An active policy expression
  - An application component
  - An application role that includes a named expression, which references the SM_USER_CONFIDENCE_LEVEL generated attribute.
- Enable confidence level support only to apply a confidence level to a realm or an application component. Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.

Follow these steps:

1. Log in to a Policy Server host system.
2. Start the XPSConfig utility.
   XPSConfig prompts for an option.
3. Enter **SM** and press Enter.
   The list of options displays and XPSConfig promtps for an option.
4. Filter the list by entering **F** and typing the string **Confidence.**
   The ConfidenceLevelSupportEnabled parameter displays.
5. Enter the number associated with the ConfidenceLevelSupportEnabled parameter to modify it.
6. Enter C to change the parameter value. The default is False.
   The pending value of the parameter changes to True.
7. Quit the XPSConfig utility.
8. Restart the Policy Server.

The confidence level support is now enabled.

CA Arcot Integration Use Cases

The following use cases detail how you can integrate CA Single Sign-On with CA Arcot strong authentication and risk evaluation. The use cases begin with a simple integration and progress into more complex scenarios.

CA Arcot Authentication and Risk Analysis

The simplest deployment includes integrating the Adapter and all related components with CA Single Sign-On.

The Adapter guides users through the authentication (CA Arcot WebFort) and risk evaluation (CA Arcot RiskFort) processes to apply a risk score during authentication (see page 267).

Follow these steps:

1. Be sure that CA Arcot RiskFort and CA Arcot WebFort are installed and configured.

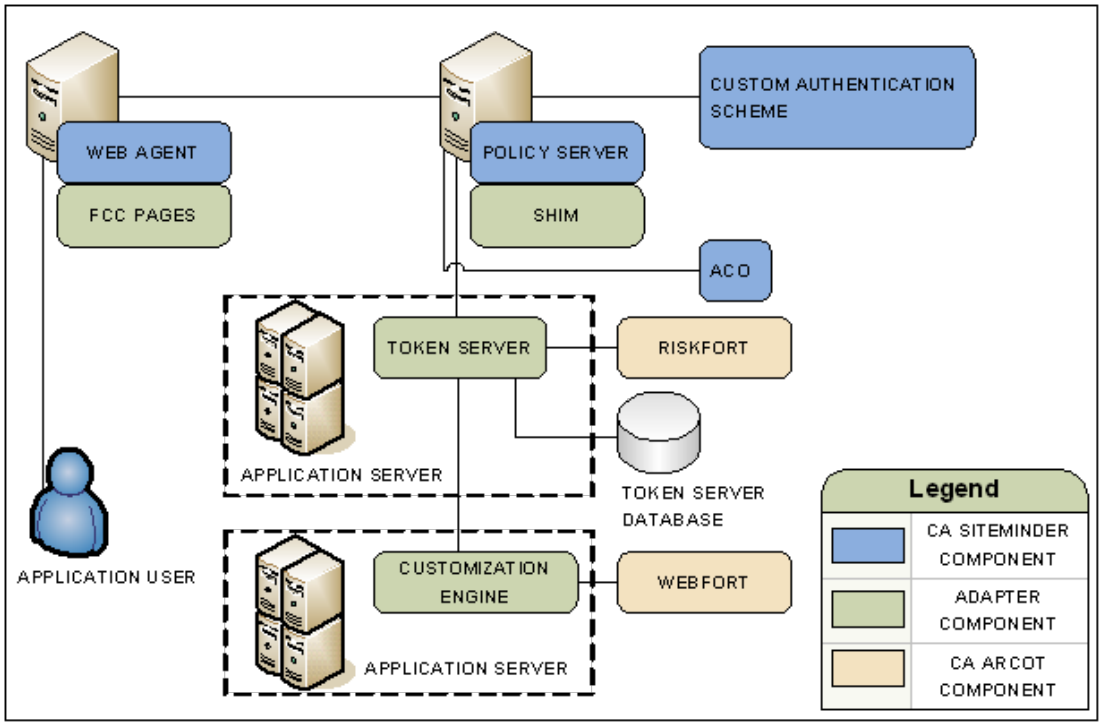
2. Install and deploy the CA Arcot Adapter and all related components. These components include a set of Forms Credential Collector files. These files let you use the Adapter HTML forms authentication scheme to gather user credentials.

3. Do the following steps:

    a. Configure a CA Single Sign-On Custom authentication scheme to call the Adapter library.

    b. Determine which Web Agents are included in the CA Arcot integration. Configure the respective Agent Configuration Objects (ACO) to support the integration.

The following diagram illustrates this deployment scenario:



SM--arcot auth and risk analysis

CA Single Sign-On Authentication and CA Arcot Risk Analysis

You can configure the Adapter for risk evaluation only by integrating a CA Single Sign-On authentication scheme. A CA Single Sign-On authentication scheme that is part of the integration is known as backing authentication.

If you use a CA Single Sign-On authentication scheme as backing authentication, the Shim acts as an interface between CA Single Sign-On and the CA Single Sign-On authentication scheme.

Note: Not all CA Single Sign-On authentication schemes are supported for backing authentication. For more information, see the CA Single Sign-On Platform Support Matrix.

Follow these steps:

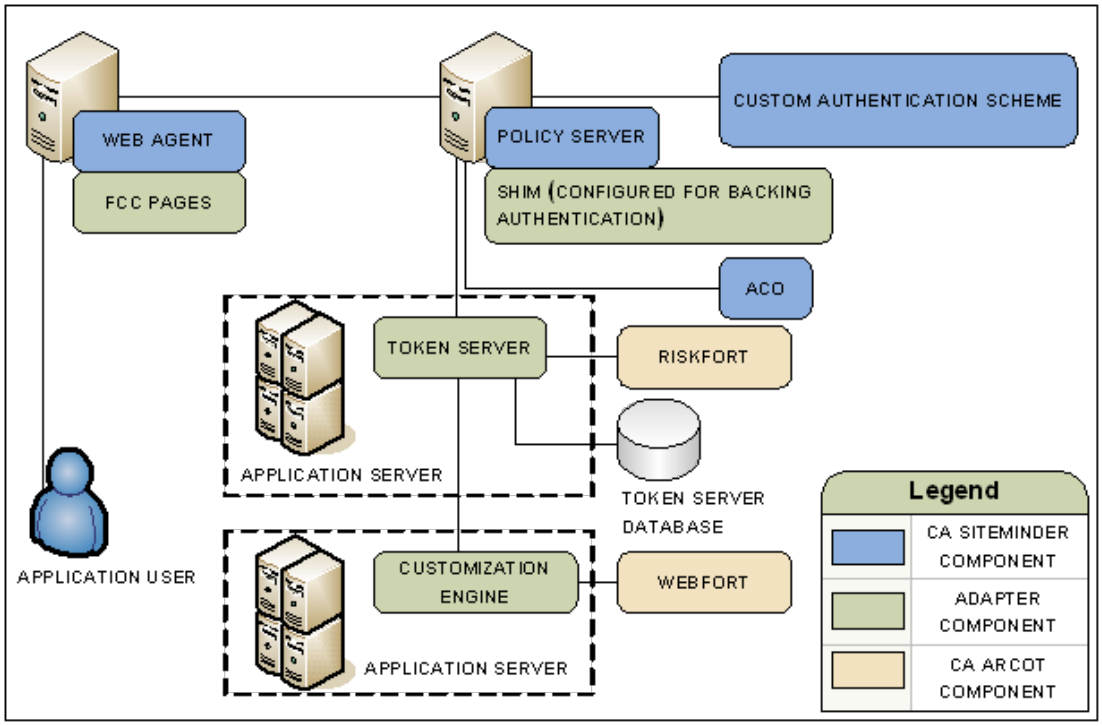
1. Complete the steps that are listed in CA Arcot Authentication and Risk Analysis.

   Important! The integration requires that a CA Single Sign-On Custom authentication scheme is configured. The CA Single Sign-On Custom authentication scheme calls the required Adapter library. This library is required even if you are deploying backing authentication.

2. Be sure that you configure the CA Single Sign-On Custom authentication scheme with a valid CA Arcot parameter. This parameter must represent a user flow that supports the CA Single Sign-On authentication scheme that is functioning as backing authentication. You enter this value in the Parameter field.

3. Configure the Shim to use the CA Single Sign-On authentication scheme as a backing authentication.

The following diagram illustrates this deployment scenario:



SM--sm auth and arcot risk analysis

CA Single Sign-On Authorization and Confidence Levels

You can extend the Policy Server authorization services by adding a confidence level (see page) to both access management models.

Adding a confidence level lets you apply the CA Arcot risk analysis results to authorization decisions.

Follow these steps:

1. Complete the steps in "CA Arcot Authentication and Risk Analysis" or " CA Single Sign-On Authentication and CA Arcot Risk Analysis" earlier in this content.

2. (Optional) If you plan on applying a confidence level to a policy realm or an application component, enable confidence level support (see page 270). Using an active policy expression or an application role to apply a confidence level remains supported from previous releases and is enabled by default.

3. Do one of the following steps:

   - If you are using policies to protect resources, add a confidence level to one or more policy realms or active policy expressions.

- If you are using applications to protect resources, add a confidence level to one or more application components or application roles.

User Store Considerations

All CA Single Sign-On users to which the integration applies must be made available to the CA Arcot WebFort database.

Contact CA Arcot Support for assistance.