CA SiteMinder®

Implementation Guide

r12.0 SP2

CA Product References

This document references the following CA products:

- CA Directory
- CA SiteMinder®

Contact CA

Contact Technical Support

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

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

Provide Feedback

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

If you would like to provide feedback about CA product documentation, complete our short customer survey, which is also available on the CA Support website, found at http://ca.com/docs.

Contents

Chapter 1: Implementation Overview

This section contains the following topics:

Purpose and Audience

This guide outlines the important architectural and configuration decisions an organization can consider when implementing SiteMinder. This document is intended to assist with the planning the following:

- A new SiteMinder implementation
- A significant modification to an existing implementation

This guide is intended for IT personnel who are familiar with the enterprise network and access management concepts and technologies.

This guide assumes familiarity with the following:

- Application servers
- Directory servers
- Databases
- Web Servers

SiteMinder Documentation

You can find complete information about the SiteMinder components, concepts, and features this guide introduces by installing the SiteMinder bookshelf (bookshelf). The bookshelf lets you:

- Use a single console to view all documents published for SiteMinder.
- Use a single alphabetical index to find a topic in any document.
- Execute a full–text search of the entire documentation set.

Note: For more information about installing the bookshelf, see the *Policy Server Installation Guide*.

More information:

Locate the Bookshelf (see page 180)

Documentation Roadmap

The bookshelf is available to help your organization complete specific tasks in a SiteMinder environment. The bookshelf contains guides in the following categories:

- Release notes
- Installation
- Configuration
- Administration
- Programming
- Federation Security Services

Release Notes

The bookshelf contains release notes. Release notes detail the enhancements and changes to major functional areas in a SiteMinder implementation. The information includes:

- New features and changes in the release
- Installation considerations and component availability
- Known issues and fixes

SiteMinder release notes include the:

- *Policy Server Release Notes*—This guide includes information for the following components:
  - Policy Server
  - All SiteMinder data stores
  - Administrative UI
  - FSS Administrative UI
  - Perl command–line interface (CLI)
- *Federation Security Services Release Notes*

- *SDK Release Notes*—This guide includes information for the following APIs:
  - C
  - Java
- *Web Agent Release Notes*

Installation

The bookshelf includes installation guides that detail the pre–requisites, concepts, and procedures for installing and configuring specific SiteMinder components:

Installation guides include the following:

- *Policy Server Installation Guide*—Use this guide to install and configure the following:
  - The Policy Server and policy store
  - Audit, key, session, and token stores

    Note: If you are trying to configure or upgrade a SiteMinder store listed in the SiteMinder Platform Support Matrix and cannot find the procedures in this guide, see the *Directory Configuration Guide*.
  - The SiteMinder FSS Administrative UI and SiteMinder Administrative UI
  - CA Business Intelligence (Report Server)
  - The OneView Monitor
  - SNMP
- *Web Agent Installation Guide*—Use this guide to install or upgrade a Web Agent.

  Note: Application server agent and ERP system agent documentation are not part of the bookshelf. These guides are available from the Download Center on the Technical Support site.
- *SiteMinder Upgrade Guide*—Use this guide for the conceptual information and procedures for planning and executing an upgrade to the latest SiteMinder release.

Configuration

The bookshelf contains configurations guides that detail the pre–requisites, concepts, and procedures for creating and managing specific SiteMinder components:

The configuration guides include the:

- *Directory Configuration Guide*—Use this guide to configure policy and user stores on a wide range of directory and database vendor products.

  Note: The *Policy Server Installation Guide* and *Policy Server Configuration Guide* also detail how to configure a policy store and user store, respectively. If you cannot find your vendor in the *Directory Configuration Guide*, refer to these guides.

- *Policy Server Configuration Guide*—Use this guide to create and manage:
  - The source of SiteMinder administrator credentials
  - Administrative user accounts for the Administrative UI
  - The infrastructure and global objects that comprise policies and applications.
- *Web Agent Configuration Guide*—Use this guide to set the default Web Agent configuration parameters to protect your Web-based resources.

Administration

Use the *Policy Server Administrator Guide* to:

- Configure and tune a SiteMinder environment at the system–level
- Monitor and maintain SiteMinder performance
- Manage users and user sessions
- Configure Policy Server audit logs
- Create SiteMinder reports

Programming

The bookshelf includes guides that explain how to use the SiteMinder APIs to integrate and extend SiteMinder capabilities in your environment.

Programming guides include the following:

- *SDK Overview Guide*—Use this guide to identify the custom applications and Policy Server extensions you can write using the APIs in the SiteMinder SDK.
- *Programming Guide for C*
- *Programming Guide for Java*

- *Programming Guide for Perl*
- *Scripting Interface for Perl*—Use this quick reference card to identify the object dependencies for the Perl CLI.

Federation Security Services

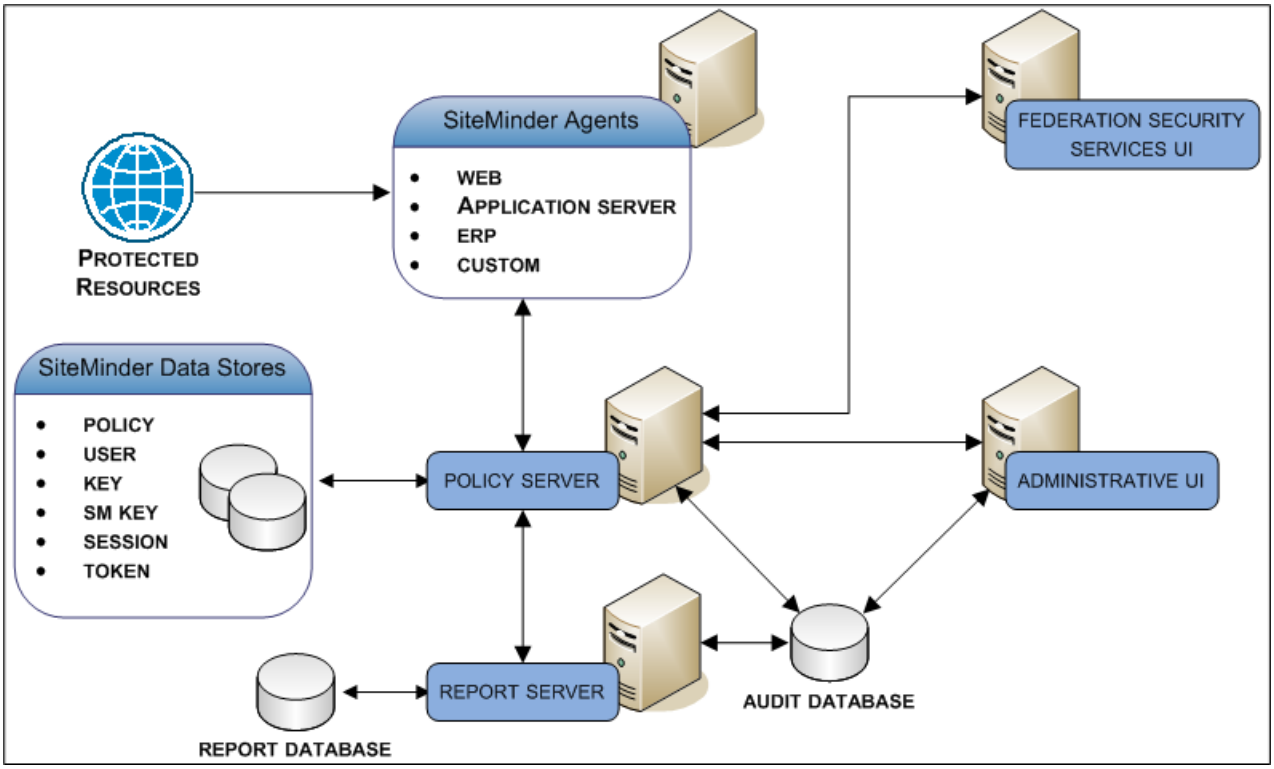
The bookshelf contains guides that detail the pre–requisites, concepts, and procedures for federating with a partner site:

- *Web Agent Option Pack Guide*—Use this guide to install or upgrade the Web Agent Option Pack.
- *Federation Security Services Overview*—Use this guide to configure the required SiteMinder components at the producing authority and consuming authority to implement a federated network.

SiteMinder Components

A SiteMinder environment includes multiple components. Some components are required to secure resources, while others are optional, or only required to implement specific features. These components work with the resources, applications, directories, and databases in your organization to provide secure access to resources in your enterprise network.

All SiteMinder components are supported on a number of operating environments. Your SiteMinder implementation is highly dependent on the environment to which you are deploying it. Your implementation does not have to reflect the following diagram. Rather, the purpose of the following diagram is to illustrate the major components in a SiteMinder environment and their general relationships with each other.



Use the previous diagram and the following component descriptions as a resource when considering the architectural questions detailed in this guide.

Policy Server

(Required) A SiteMinder Policy Server (Policy Server) acts as the Policy Decision Point (PDP). The purpose of the Policy Server is to evaluate and enforce access control policies, which it communicates to a SiteMinder Agent. A Policy Server provides the following:

- Policy-based user management
- Authentication services
- Authorization services
- Password services

- Session management
- Auditing services

The Policy Server interacts with all other major components to perform these tasks.

More information:

Documentation Roadmap (see page 10)

Agents

(Required) A SiteMinder Agent can reside on a web server, a J2EE application server, an Enterprise Resource Planning (ERP) system, or custom application. An Agent acts as the Policy Enforcement Point (PEP), intercepting user requests for resources and communicating with a Policy Server to determine if the resource is protected.

If the resource is not protected, the Agent allows access. If the resource is protected, the Agent continues to communicate with the Policy Server to authenticate and authorize users. A successful authorization prompts the Agent to let the resource request proceed to the server. Agents also:

- Provide information to web applications to enable content personalization
- Cache information about authenticated users and protected resources to allow quicker access to resources
- Enable single sign–on (SSO)

More information:

Documentation Roadmap (see page 10)

CA Business Intelligence

(Optional) CA Business Intelligence is a set of reporting and analytic software that various CA products use for the purposes of presenting information and supporting business decisions. CA products use CA Business Intelligence to integrate, analyze, and then present, through various reporting options, information required for effective enterprise IT management.

Included in CA Business Intelligence is BusinessObjects Enterprise XI 2.1, a complete suite of information management, reporting, and query and analysis tools. CA Business Intelligence installs BusinessObjects Enterprise XI as a stand–alone component. In this guide, this stand–alone component is referred to as the Report Server. Installing the Report Server is a separate step within the overall SiteMinder installation process. Installing the Report Server separately from SiteMinder–specific components lets other CA products share the same Business Intelligence Services.

The Report Server compiles reports to help you analyze your SiteMinder environment. The purpose of this component it to create the following types of reports:

- Audit
- Policy analysis

The Report Server communicates with the following components to compile reports:

- The Central Management Server (CMS) database (report database)
- An Administrative UI
- A Policy Server
- A SiteMinder audit database

More information:

Documentation Roadmap (see page 10)

Data Stores

A SiteMinder implementation contains multiple data stores. Some stores are required, while others are optional, or only required to implement specific features.

The following descriptions detail:

- If the store is required or optional
- The purpose of the store

Policy Store

(Required) The SiteMinder policy store (policy store) is an entitlement store that resides in an LDAP directory server or ODBC database. The purpose of this component is to store all policy-related objects, including the:

- Resources SiteMinder is protecting
- Methods used to protect those resources
- Users or groups that can or cannot access those resources
- Actions that must take place when users are granted or denied access to protected resources

The Policy Server uses this information, collectively known as a policy, to determine if a resource is protected and if an authenticated user is authorized to access the requested resources.

Note: For more information about configuring a policy store, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

User Store

(Required) A SiteMinder user store connection (user store connection) is a connection to an existing user directory or database in your enterprise network. You are not required to use a proprietary SiteMinder user store. The purpose of the user store connection is to make user data available to the Policy Server, which includes the following:

- Organizational information
- User and group attributes
- User credentials, such as passwords
- User attributes, such as first and last name

The Policy Server uses these connections to:

- Verify user credentials when an Agent submits a request for a protected resource
- Retrieve user attributes for the SiteMinder features that require specific user data

Note: For more information about configuring a user store connection, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

External Administrative User Store

(Optional) By default, the Administrative UI uses the policy store as its source for SiteMinder administrator credentials. This default configuration lets you manage the environment immediately after configuring a policy store and installing the Administrative UI. When you configure a policy store, the default SiteMinder super user account (siteminder) is created. This account has maximum system privileges, and is used to access the Administrative UI for the first–time and to create additional SiteMinder administrators.

You can configure the Administrative UI to use an external administrator user store, for example, a corporate directory. An external administrative user store is a connection to an LDAP directory server or ODBC database in your enterprise network. Consider the following:

- An Administrative UI can only connect to a single external administrative user store.
- An Administrative UI can be configured to managed multiple Policy Servers. If an Administrative UI is to manage multiple Policy Servers, a connection to an external administrator user store is required.
- If you configure more than one Administrative UI for high–availability, the same external administrative user store makes all administrators available to each Administrative UI.

Note: For more information about SiteMinder administrators and configuring an external administrative user store, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

Key Store

(Required) By default, the SiteMinder key store (key store) is located with the policy store. The purpose of this component is to store the encryption keys Policy Servers and Agents use to encrypt sensitive data, which include:

- The keys Agents use to encrypt SiteMinder cookies.
- The keys Policy Servers use to encrypt sensitive policy store information, such as administrator passwords.
- The keys Policy Servers use to encrypt SiteMinder session tickets that contain credentials and other information related to user sessions.

You can store encryption keys in a separate directory or database. The need to deploy a separate key store depends on:

- How you implement Policy Servers and policy stores
- Your single sign–on requirements

Note: For more information about configuring a key store, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

Token Store

(Optional) SiteMinder supports several authentication schemes which use third–party hardware-based security cards or tokens. These tokens provide dynamically–generated passwords that add another layer of security.

All tokens use data files provided by their respective vendors. Some vendors store their tokens remotely on a server they provide. Other vendors allow their tokens to be stored locally in the SiteMinder token store (token store). The purpose of this component is to store these tokens.

If you decide to use a token–based authentication scheme, the token vendor you implement influences if a token store is a required component in your SiteMinder environment.

Consider the following if a token store is a required component in your implementation:

- You can configure a token store to a stand–alone database.
- If the policy store is configured in a database that is also supported for use as a token store, you can configure the policy store to function as a token store.

Note: For more information about configuring a token store, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

SiteMinder Audit Database

(Optional) By default, the Policy Server writes audit events to a text file, which is known as the Policy Server log. The purpose of audit logs is to track information about all user activity, including:

- All successful authentications
- All failed authentications
- All successful authorization attempts
- All failed authorization attempts
- All administrative login attempts
- All administrative actions, such as changes to administrator passwords, the creation of policy store objects, and changes to policy store objects

However, you can configure a stand–alone SiteMinder audit database (audit database). When deciding where to store audit events, consider that:

- The Report Server requires a connection to an audit database to create audit–based reports. The Report Server cannot create audit–based reports from a Policy Server log written to a text file.
- Storing audit logs to a database is more secure than logging the information to a text file.
- If supported, a policy store can also function as an audit database.

Note: For more information about configuring an audit database, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

Session Store

(Optional) When SiteMinder authenticates a user, the Policy Server issues a session ticket. A session ticket contains basic information about the user and authentication information for the user. By default, SiteMinder implements session management through non–persistent sessions. If non–persistent sessions are enabled, an Agent writes the session ticket to a cookie on the browser of the users. However, some SiteMinder features require persistent sessions.

If persistent sessions are enabled, an Agent must write the session ticket to a stand–alone database.

You deploy a SiteMinder session store (session store) for the following primary reasons:

- If a SiteMinder log off URI is implemented, a session store prevents a SiteMinder session from being used again after a user logs off.
- To provide support for features that require persistent user sessions.

Agents use this information to identify users and provide session information to the Policy Server.

Note: For more information about configuring a session store, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

SiteMinder Key Database

(Optional) Some SiteMinder features and some SiteMinder product integrations require keys for signing, verifying, encrypting, and decrypting information. The purpose of the SiteMinder key database (smkeydatabase) is to make the private keys and the certificate authority (CA) certificates available to the environment.

The smkeydatabase is a separate component from a SiteMinder key store, and is a local, embedded database that does not require an external store.

For example, use of the SiteMinder Microsoft InfoCard authentication scheme and integrating SiteMinder with Federation Security Services are both examples of an implementation that require a smkeydatabase.

Note: For more information about a smkeydatabase, see the documentation roadmap.

More information:

Documentation Roadmap (see page 10)

Administrative User Interfaces

A SiteMinder implementation can contain two Administrative User Interfaces (UI). The following descriptions detail:

- If the UI is required
- The purpose of the UI

SiteMinder Administrative UI

(Required) The SiteMinder Administrative UI (Administrative UI) is a web-based administration console that is installed independent of the Policy Server. The Administrative UI functions as the primary UI in a SiteMinder implementation and is intended for managing all tasks related to access control, such as:

- Authentication and authorization policies
- Enterprise Policy Management (EPM)
- Reporting and policy analysis

The Administrative UI is intended for viewing, modifying, and deleting all Policy Server objects, except objects related to Federation Security Services (FSS). All federation-related configuration tasks are managed using the FSS Administrative UI.

More information:

Documentation Roadmap (see page 10)

Federation Security Services Administrative UI

(Optional) The SiteMinder Federation Security Services Administrative UI (FSS Administrative UI) is an applet–based application that is optionally installed with the Policy Server. Federation Security Services components consist of the affiliates (consumers, service providers, resource partners) and SAML authentication schemes that you configure to support federated communication between two partners.

The FSS Administrative UI is intended for only managing tasks related to SiteMinder Federation Security Services.

More information:

Documentation Roadmap (see page 10)

Chapter 2: Architectural Considerations

This section contains the following topics:

Your Enterprise Environment

SiteMinder implementations are highly dependent on the environments in which you deploy them. We recommend that you develop a plan that breaks your implementation into steps that make sense for your enterprise. As you plan the deployment, there are many questions for consideration.

The answers to these questions are critical to planning your SiteMinder implementation.

Operating Systems

SiteMinder components are supported across multiple platforms, the details of which are located in the SiteMinder Platform Support Matrix. Which of the following operating systems has your enterprise deployed?

- Microsoft® Windows®
- Sun™ Solaris™
- Red Hat® Enterprise Linux®
- Novell® SUSE® Linux
- Hewlett–Packard Company UNIX (HP–UX)
- IBM® AIX®
- IBM z/OS®

Note: For the specific versions of supported operating systems, see the SiteMinder Platform Support Matrix.

Use the following table to help determine if the operating systems your enterprise has deployed are currently supported for the required SiteMinder components.

Component	Required?	Operating Systems
Policy Server	Yes	
Agent	Yes	
Administrative UI	Yes	
FSS Administrative UI	No	
Report Server	No	

Note: Additional non–platform requirements exist for each of these components, for example, minimum memory requirements. For more information about non–platform requirements for the Policy Server, the Administrative UI, the FSS Administrative UI, and the Report Server, see the *Policy Server Installation Guide*. For more information about non–platform requirements for an Agent, see the specific SiteMinder Agent documentation.

Web Server Vendors

You can install and configure SiteMinder Agents to protect resources on web servers. Which of the following supported server vendors has your enterprise deployed?

- Apache™ HTTP Server
- Apache Tomcat
- Hewlett–Packard Company (HP) Apache
- IBM HTTP Server
- IBM Lotus® Domino
- Microsoft IIS
- Oracle® HTTP Server
- Red Hat Apache
- Sun Java™ System

Note: For the specific versions of the supported web servers, refer to the SiteMinder Platform Support Matrix.

If you use other web servers not listed here, consider using the Secure Proxy Server to protect the resources on these web servers.

Application Server Vendors

You can install and configure Agents to protect resources on J2EE application servers. Which of the following supported server vendors has your enterprise deployed?

- Oracle WebLogic®
- IBM WebSphere®

Note: For the specific versions of supported application servers, refer to the SiteMinder Platform Support Matrix.

Enterprise Resource Planning Systems

You can install and configure Agents to protect resources on ERP systems. Which of the following supported ERP vendors has your enterprise deployed?

- Oracle PeopleSoft®
- Oracle Siebel®
- SAP®

Note: For the specific versions of supported ERP systems, see the SiteMinder Platform Support Matrix.

Directory Servers and Databases

SiteMinder data stores are supported across multiple directory servers and databases. Which of the following vendors has your enterprise deployed?

- CA Directory
- Critical Path InJoin Directory Server
- IBM DB2®
- IBM Directory Server
- IBM Lotus® Domino® LDAP
- Microsoft Active Directory®
- Microsoft ADAM
- Microsoft SQL Server®
- MySQL®
- Novell® eDirectory™
- Oracle Internet Directory
- Oracle RDBMS
- Oracle RAC
- Oracle LDAP
- Radiant Logic, Inc. RadiantOne™ Virtual Directory Server
- Red Hat Directory Server
- Siemens® DirX® Directory
- Sun Java™ System Directory Server
- Sun ONE Directory Server

Use the following table to help determine if the directory server and database types your enterprise has deployed are currently supported for the SiteMinder components your implementation requires.

Component	Required	LDAP?	Database?
Policy store	Yes		
User store connection	Yes		
Administrative user store	No		
Audit database	No	N/A	
Key store	No		
Token store	No	N/A	
Session store	No	N/A	

Note: For the specific versions of supported directory servers and databases, see the SiteMinder Platform Support Matrix.

More information:

Locate the SiteMinder Platform Support Matrix (see page 179)

Implementation Considerations

The decisions related to how you implement SiteMinder depend on:

- How you map your applications to the SiteMinder access management models
- The SiteMinder features you plan to use
- How you plan to manage SiteMinder Policy Servers and Web Agents

We recommend that you consider the following before you deploy and configure SiteMinder.

Access Management Models

SiteMinder 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 SiteMinder grants access to the resource.
- What happens when SiteMinder denies access to a resource.

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

- A SiteMinder policy
- An Enterprise Policy Management (EPM) application

Note: The following SiteMinder core objects are required to configure a SiteMinder policy or an EPM application:

- A host configuration object
- An agent configuration object
- An agent object
- A user directory object

  For more information about these objects, see the Policy Server Configuration Guide.

SiteMinder Policy

Before the release of SiteMinder r12.0, managing SiteMinder policies was the only way to protect resources. A SiteMinder policy is a SiteMinder–centric access management model. A SiteMinder policy is comprised of individual SiteMinder core objects, which include the following:

- 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 SiteMinder policy object binds these core objects to identify resources, user populations, and the required actions when SiteMinder grants or denies access to the resource. As such, configuring a SiteMinder policy requires an understanding of each object.

Note: For more information about each of these objects and their individual SiteMinder policy roles, see the Policy Server Configuration Guide.

EPM Application

SiteMinder r12.0 introduced the EPM application. EPM is an application–centric access management model. EPM presents access management in the context of securing an application.

To protect an application, you are only required to provide data for configuration settings that do not have defaults. Modifying other settings is optional, and although not required, you can manage additional SiteMinder settings to modify EPM settings beyond the default settings to define more fine–grained protection.

If you are familiar with the core SiteMinder objects, there is a relationship between the application–oriented concepts and the underlying SiteMinder components. The following table summarizes this relationship.

Application Dialogs and Group Boxes	Underlying SiteMinder Component
General settings	Defines the SiteMinder policy domain and the root location of the protected resources.
Components	Defines the realm and the location of the resources within the application that share the same security requirements.
Resource	Specifies the rule and the required authentication or authorization actions.
Application Roles	Replaces the function of user directory lookups.

Unlike a SiteMinder policy object, you do not have to create the individual domain, realm, and rule objects. When you create the application, SiteMinder creates the objects automatically and binds them to identify resources, user populations, and the required actions when SiteMinder grants or denies access to the resource. As such, configuring an application does not require an understanding of these core objects.

Note: For more information about EPM applications, see the Policy Server Configuration Guide.

Identify the Applications to Secure

Which applications are you planning to secure? How do they map to the SiteMinder 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 SiteMinder policy domain or the resource filter of an EPM application. A SiteMinder 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 SiteMinder policy realm or an EPM application component.

Grouping resources in this way helps you map applications to the SiteMinder 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.

More information:

Capacity Planning Introduced (see page 75)
Ignore Extensions Parameter (see page 127)

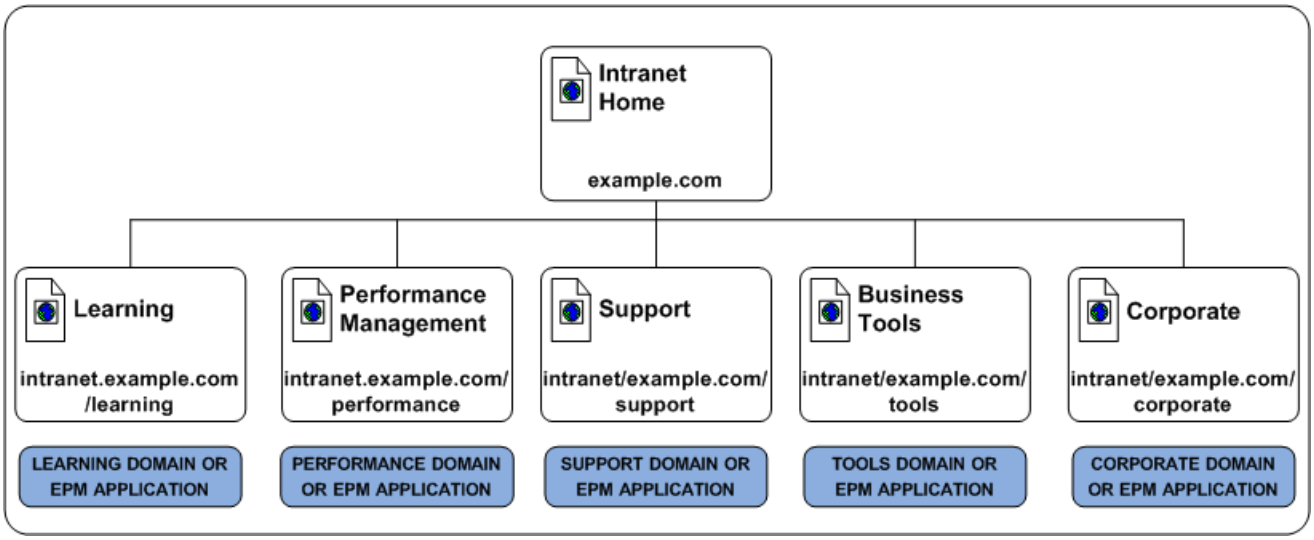
Group Resources into Domains or EPM Applications

Defining a SiteMinder 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.

Note: For more information about a SiteMinder policy domain or an EPM application, see the *Policy Server Configuration Guide*.

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

For example, a fictitious company, 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

More information:

Domains and Authentication Performance (see page 137)

Group a Resources into Realms or EPM Components

Defining a SiteMinder policy realm or an EPM component depends on identifying sets of individual resources (URLs) that share the same security or personalization requirements within a SiteMinder 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.

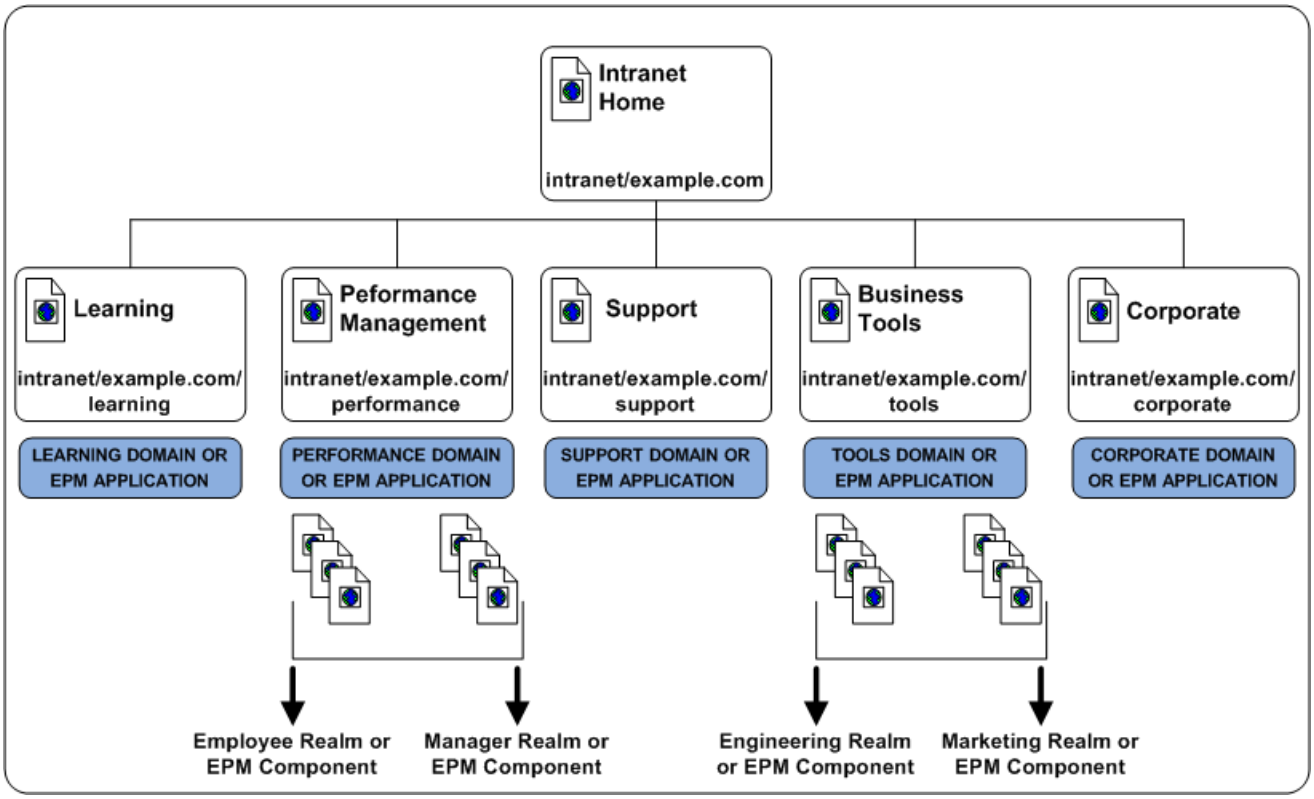
Note: For more information about SiteMinder policy realms and EPM components, see the Policy Server Configuration Guide.

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 SiteMinder 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, SiteMinder policy rules and EPM resources are used to control specific Web Agent, authentication, and authorization events. For more information, see the Policy Server Configuration Guide.



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
Business Tools	intranet.example.com/tools	/engineering

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

Identify User Stores

SiteMinder 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 31), 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 SiteMinder Administrator must configure in a SiteMinder policy domain to protect the resource.

  Note: For more information about configuring user store connections within a domain, see the Policy Server Configuration Guide.

- Determine if your environment requires the SiteMinder directory mapping feature. By default, SiteMinder assumes that users are authenticated and authorized against the same user store or stores. However, you can configure a SiteMinder policy domain to authenticate against one or more stores and authorize against others.

  Note: For more information about directory mapping, see the Policy Server Configuration Guide.

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

SiteMinder supports multiple authentication methods to meet the varying levels of protection your resources require:

- Basic
- Forms-based user ID and password
- Hardware and software token-based, such as RSA® Ace/SecurID®
- Integrated Windows Authentication (IWA)
- Information Card Authentication Schemes (ICAS), such as Microsoft Windows CardSpace
- MIT Kerberos
- Server-based, such as RADIUS and SafeWord
- X.509 Certificate-based
- Custom Authentication schemes created using the SiteMinder SDK

After you identify the applications to secure (see page 31), 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 SiteMinder Administrator must configure to protect a specific resource.

  Note: For more information about configuring authentication schemes, see the Policy Server Configuration Guide.

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 31).

Resource	URL	Realm	Authentication Method

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 SiteMinder 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.

Note: For more information, see the *SiteMinder Policy Server Configuration Guide*.

Password Policy Considerations

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

- SiteMinder needs read and 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 SiteMinder 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 SiteMinder password policy or
  - Disabled

  Otherwise the native password policy accepts or reject passwords without notifying SiteMinder. Consequently, SiteMinder cannot manage those passwords.
- If you use password policies on multiple Policy Servers, the system times of all the servers must be synchronized to avoid disabling accounts or forcing password changes prematurely.

  Note: For more information, see the *SiteMinder Policy Server Configuration Guide*.

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 SiteMinder Web Agents. Consider the following questions:

1. Are most of your web applications deployed on a large server farm with the same security requirements?
2. Are most of your web applications managed by a centralized person or group?
3. Are most of your web applications deployed on separate web servers with different security requirements?
4. 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 one or more Web Agents from an Agent Configuration Object (ACO) that resides in the Policy Server. With central configuration, you can update the parameter settings of several Web Agents at once. Generally, each distinct web application uses a separate ACO, whose settings are shared among all the Web Agents that protect the web application. For example, if you have five Web Agents protecting one accounting application, you can create one ACO with the settings for the application. All five Web 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 a Web Agent starts, it reads the value of the AllowLocalConfig parameter of its associated ACO. If the value is set to no, then the Web Agent uses the parameter settings from the ACO.

Note: We recommend using central web 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.

Note: For more information about the location of the LocalConfig.conf file on your respective web server, see the *Web Agent Configuration Guide*.

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.

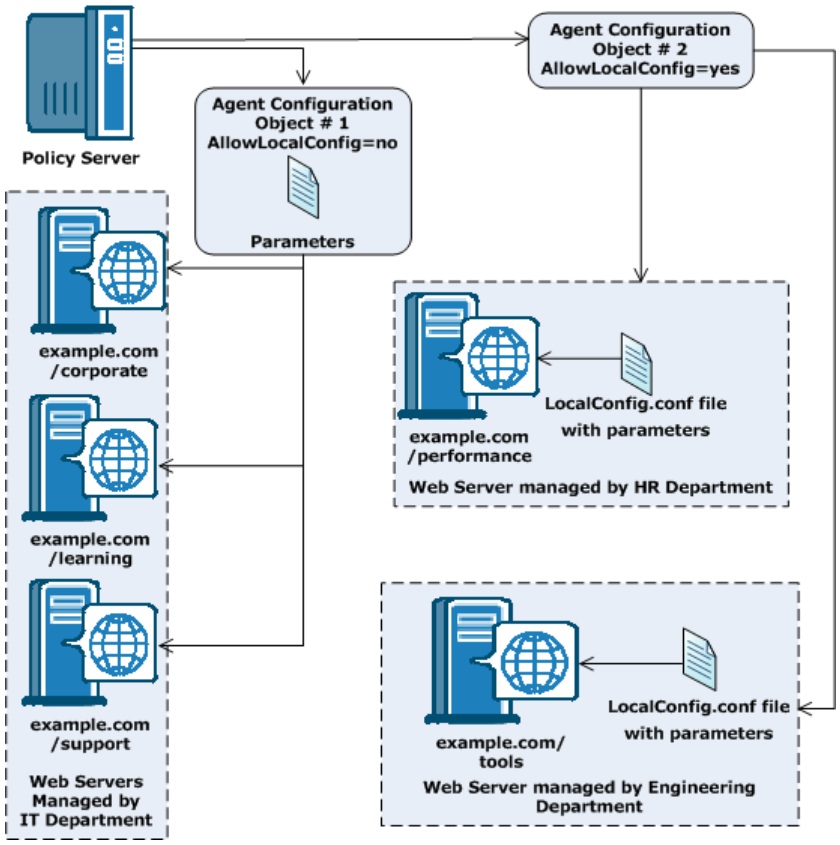
- Do you want to allow local web server administrators to change some Web Agent configuration settings but not others?

  For example, your organization uses SiteMinder 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.

  Note: For more information, see the *Web Agent Configuration Guide*.

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:



Identify Data Centers

Multiple factors, which are discussed later, can influence how you decide to implement SiteMinder components across multiple data centers. Identifying the data centers and the purpose each is to serve in your SiteMinder environment prepares you to make informed decisions when determining how to implement SiteMinder 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

More information:

Multiple Data Centers (see page 93)

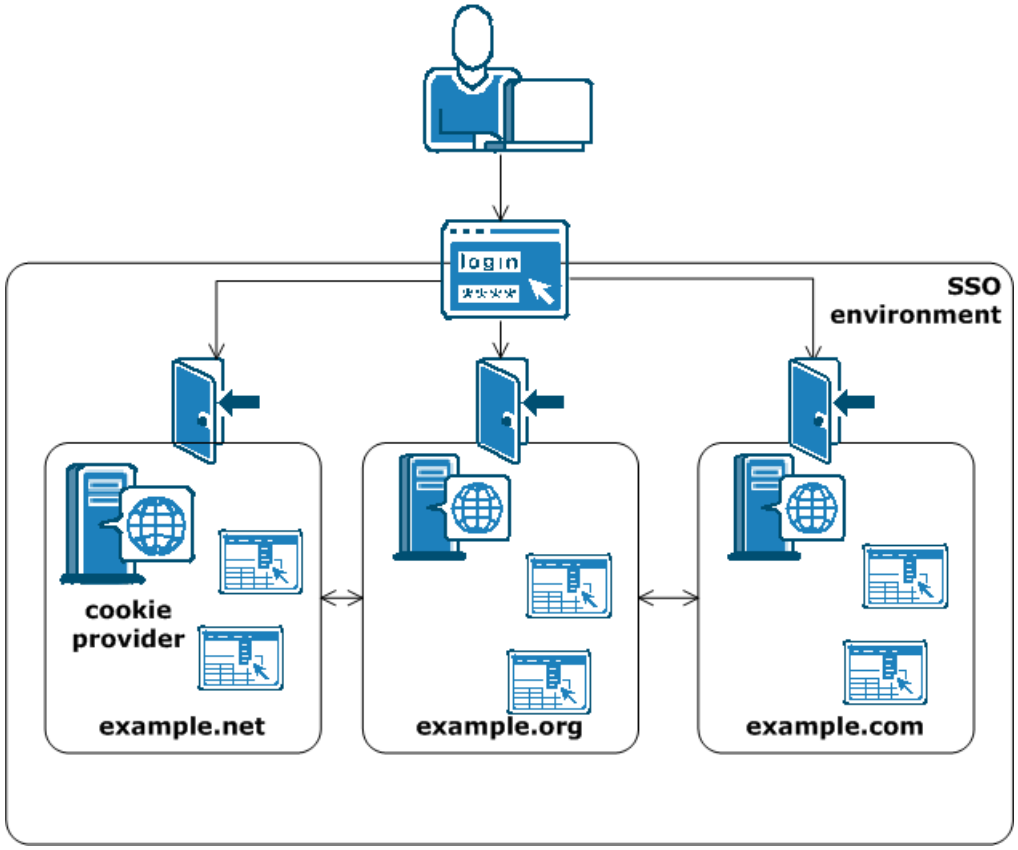
Identify Resources to be Secured with Multiple Cookie Domains

Will the single-sign on environment in your enterprise extend across multiple cookie domains?

SiteMinder implements single sign-on across multiple cookie domains using a SiteMinder Web Agent configured as a cookie provider.

The cookie domain where the cookie provider Web Agent resides is named the cookie provider domain. All the other Web Agents from the other cookie domains within the single sign-on environment, point to one cookie provider.

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

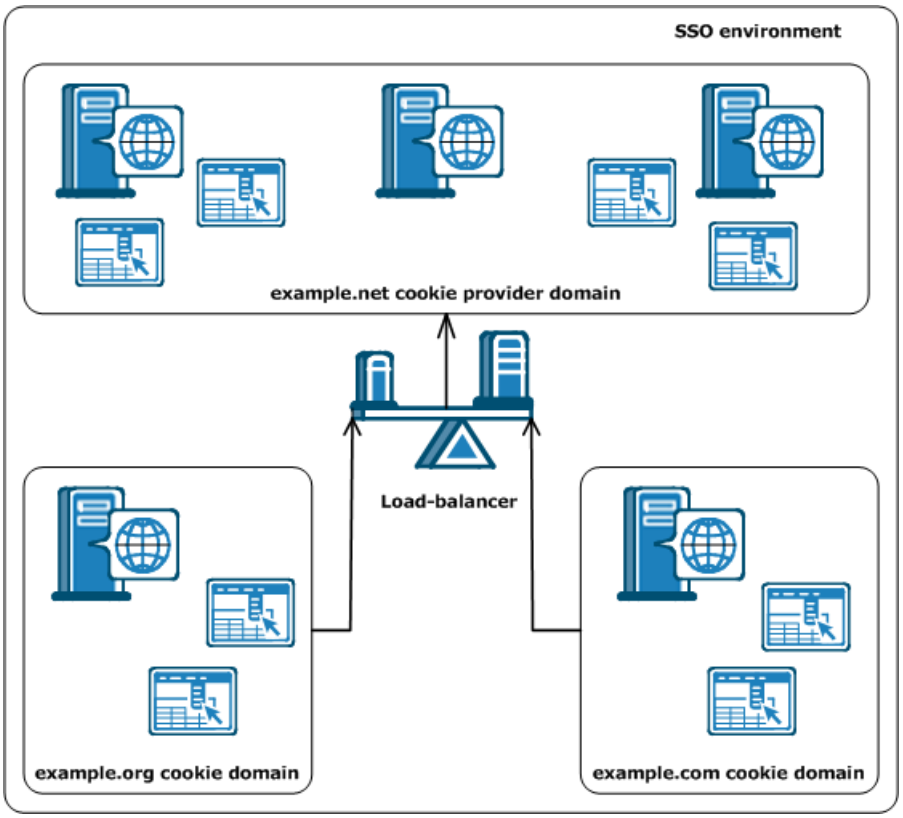


Note: For more information about cookie providers, see the *Web Agent Configuration Guide*.

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

Will the Web Agents in your single sign-on environment need to be load-balanced?

Because all Web 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 as shown in the following illustration:



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 Federation Security Services

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

CA SiteMinder Federation Security Services lets you extend SiteMinder functionality to partner sites by enabling identity federation. Federated transactions between partner organizations let your enterprise:

- Exchange user 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 received from a partner
- Allow interoperability across heterogeneous environments

Federation Security Services lets your enterprise generate, consume, or generate and consume assertions. Federation Security Services supports the following communication standards and protocols:

- SAML 1.0, 1.1, and 2.0
- Microsoft ADFS/WS-Federation
- SAML browser artifact protocol
- SAML POST protocol
- WS-Federation Passive Requestor Profile protocol

Note: Federation Security Services is separately licensed from SiteMinder. Contact your CA account representative for more information about licensing. For more information about Federation Security Services, see the *Federation Security Services Guide*.

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

Partner	Standard	Protocol

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 SiteMinder 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 SiteMinder environment only uses AES–compliant algorithms to encrypt sensitive data. A SiteMinder environment can operate in one of the following FIPS modes of operation.

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

Note: For more information about the cryptographic libraries SiteMinder uses and the AES algorithms used to encrypt sensitive data in FIPS–only mode, see the *Policy Server Administration Guide*. For more information about the FIPS modes of operation and which to use when installing the Policy Server, see the *Policy Server Installation Guide*.

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 SiteMinder 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 SiteMinder. 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 r12.0 SP2 versions of the respective SDKs to achieve the required support for FIPS–only mode.

Determine if Virtualization is to be Used

Will SiteMinder be implemented to a virtual environment?

Consider the following before implementing SiteMinder to a virtual environment:

- Be sure to review the CA policy on virtualization.
- 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 detailed in each SiteMinder installation guide 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 SiteMinder 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 SiteMinder 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 SiteMinder 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

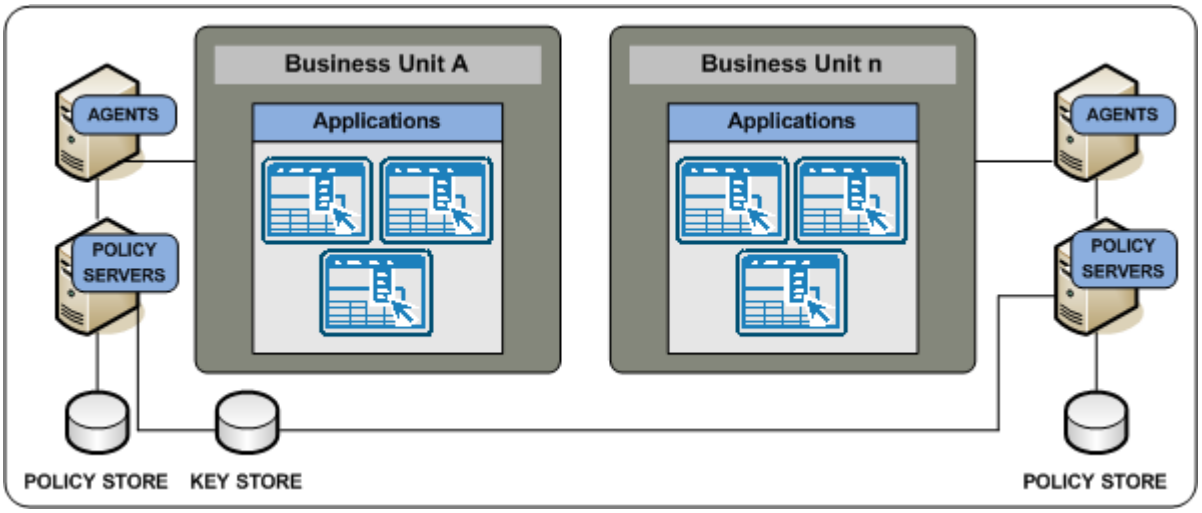
If individual business units manage Policy Servers and policy stores locally, consider that local Policy Server management:

- Lets each business unit manage their security requirements based on their individual needs.
- Can increase the complexity of the SiteMinder 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 SiteMinder configuration. As illustrated, Policy Servers in both business units must share a key store to let all SiteMinder 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. For more information about key management scenarios to facilitate single sign–on, see the *Policy Server Administration Guide*.

- Can make a consistent implementation and management of SiteMinder core objects, policies, and EPM applications more challenging because SiteMinder administrators are located in disparate business units.

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



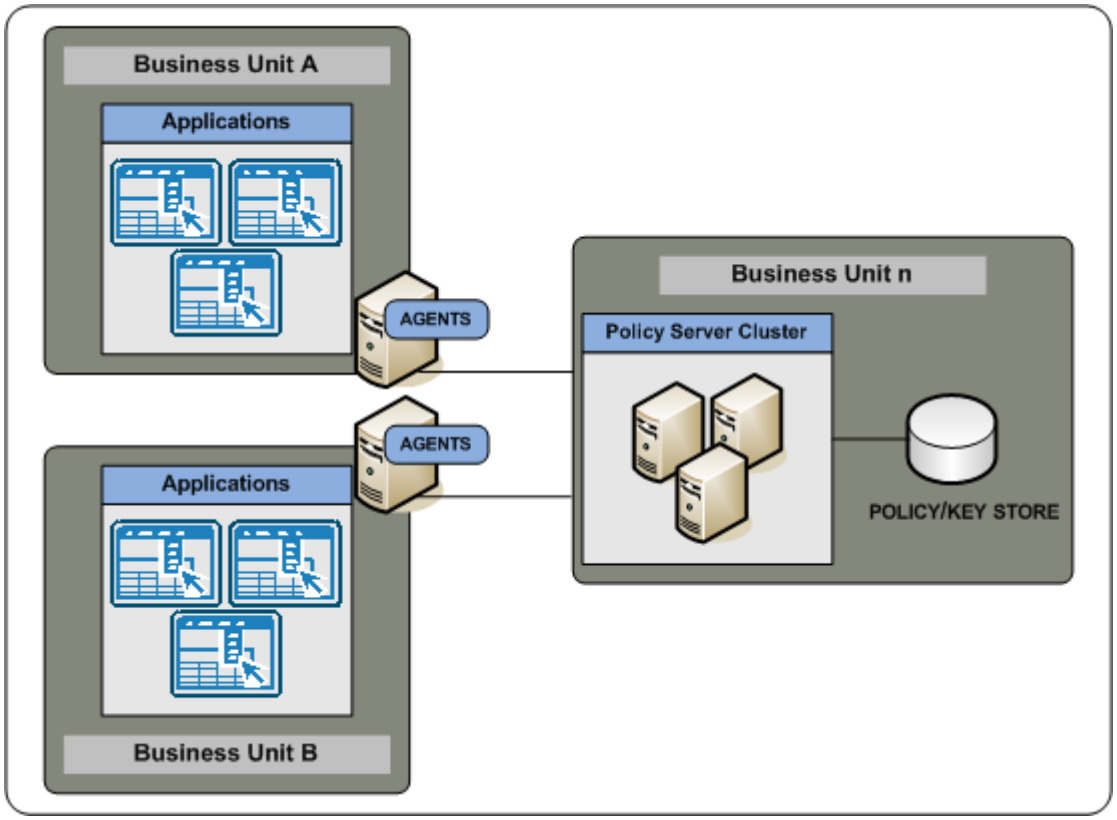
Central Policy Server Management

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

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

  Note: As illustrated, individual business units can continue to manage the SiteMinder Agents protecting their applications.
- Can simplify the SiteMinder infrastructure. Central management can result in fewer Policy Server and policy stores to manage and upgrade.
- Lets administrators monitor SiteMinder 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.

Note: For more information, see the *Web Agent Configuration Guide*.

Architectural Use Cases

The purpose of the following use cases is to get you thinking about your SiteMinder 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 SiteMinder 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 SiteMinder 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

More information:

Implementation Considerations (see page 29)
Redundancy and High Availability (see page 60)
Capacity Planning Introduced (see page 75)
Performance Tuning Introduced (see page 111)

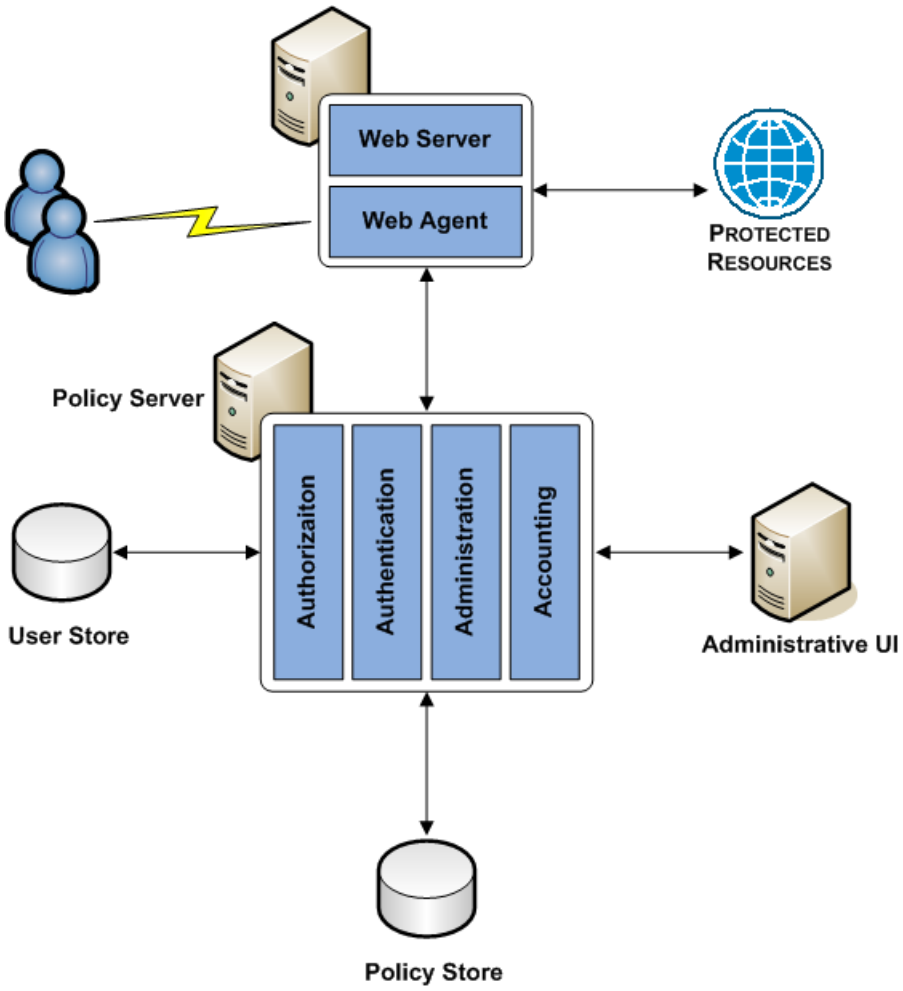
Simple Deployment

The simplest SiteMinder 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 SiteMinder Components.

More information:

SiteMinder Components (see page 13)

Simple Deployment with Optional Components

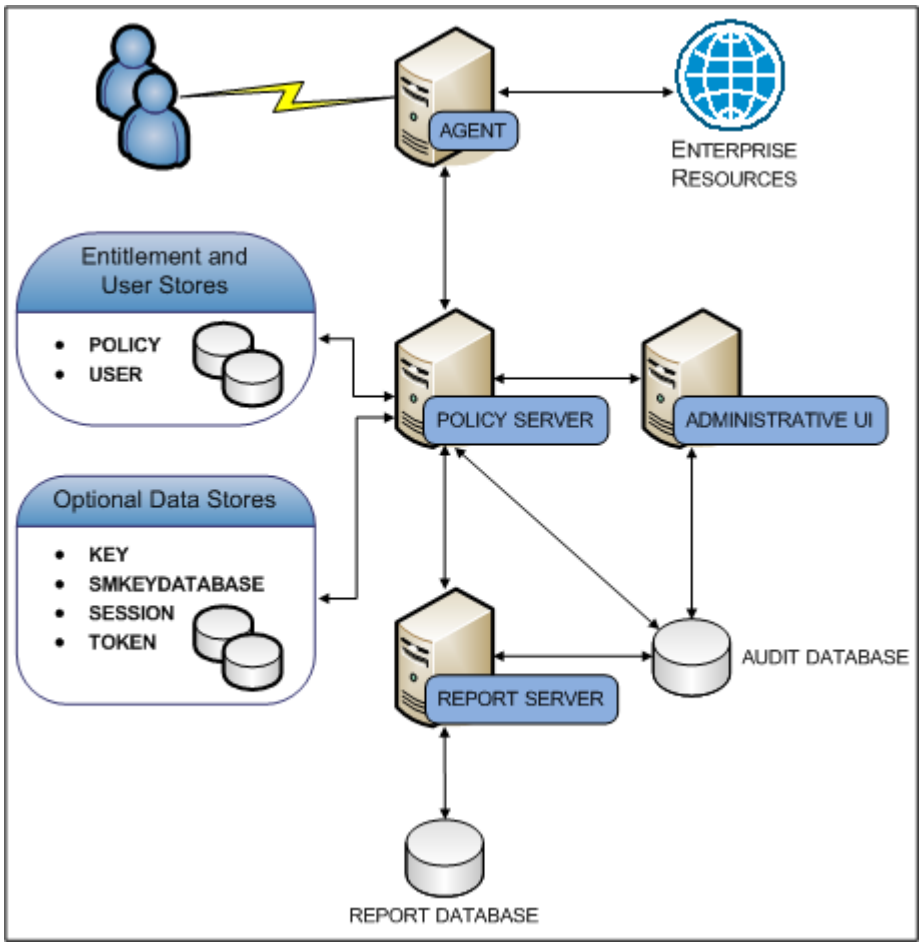
You can extend the functionality of a simple deployment through the use of optional SiteMinder components. The decision to implement optional components is determined by the SiteMinder features your enterprise requires. For example:

- If you are planning to implement Federation Security Services, your environment requires a smkeydatabase and a session store.
- If you are planning to implement a token-based authentication scheme, your environment may require a token 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

- An smkeydatabase
- A token store



Each component has a specific role in resource protection.

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

More information:

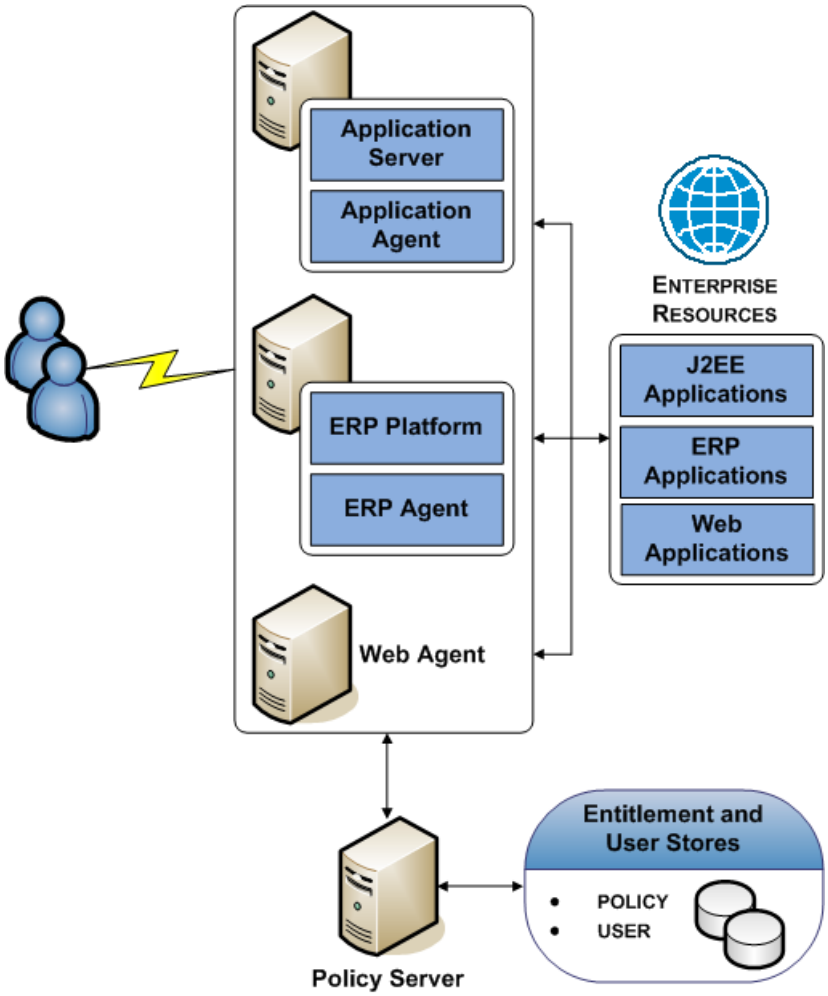
SiteMinder Components (see page 13)

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 SiteMinder Components.

More information:

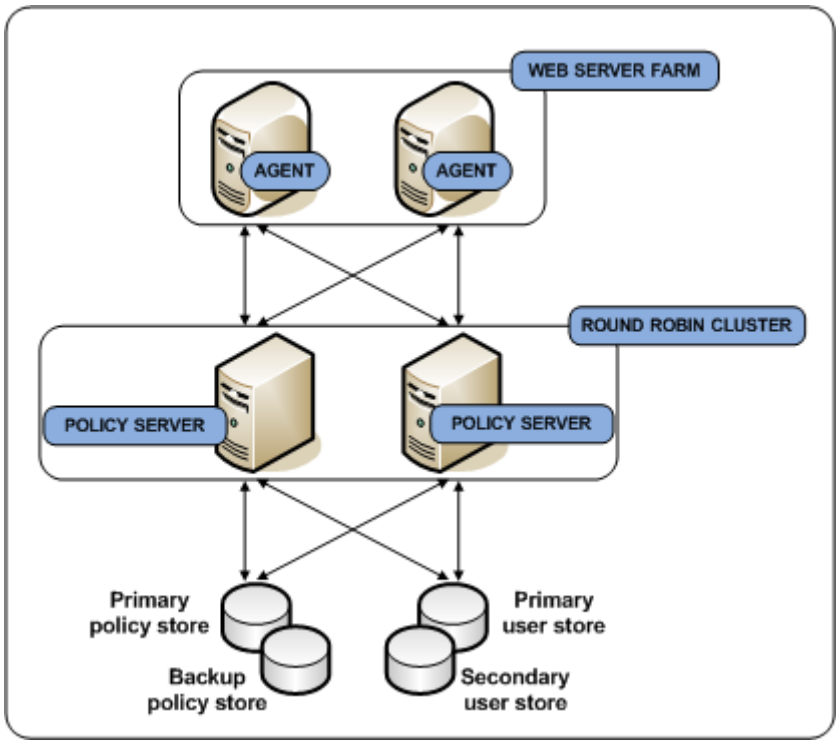
SiteMinder Components (see page 13)

Multiple Components for Operational Continuity

You can implement multiple blocks of components to build redundancy and failover into the environment. 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 SiteMinder Components. For more information about SiteMinder redundancy and high availability, see Redundancy and High Availability.

More information:

SiteMinder Components (see page 13)
Redundancy and High Availability (see page 60)

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:

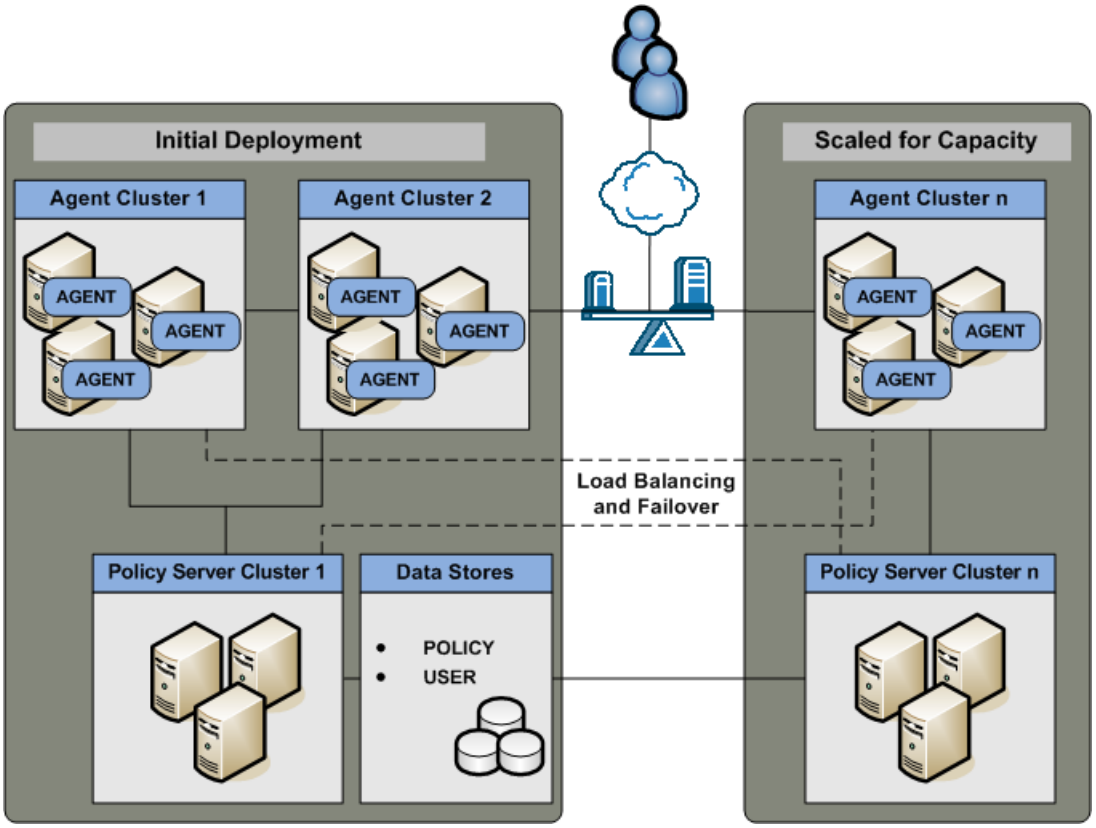
- A load balancer distributing user requests across multiple Agent clusters.
- Multiple Agent instances intercepting user requests for specific applications. Agents are configured to initialize and communicate with a primary Policy Server in the cluster. If enough Policy Servers in the cluster become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

  Note: For more information about failover thresholds for Policy Server cluster failover thresholds, see the *Policy Server Administration Guide*.

More information:

SiteMinder Components (see page 13)
Redundancy and High Availability (see page 60)

Redundancy and High Availability

You configure redundancy and high availability between logical blocks of SiteMinder components to maintain system availability and performance.

Agent to Policy Server Communication

When you configure a SiteMinder Web Agent, a Host configuration file (named SmHost.conf by default), is created on the web server. The Web Agent uses the connection information in this Host configuration file to create an initial connection with a Policy Server.

Note: For more information about installing a Web Agent, see the *Web Agent Installation Guide*.

After the initial connection is established, the Web 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 how the Web Agent is to distribute requests among multiple Policy Servers.

A Web 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

More information:

Agents (see page 15)

Failover

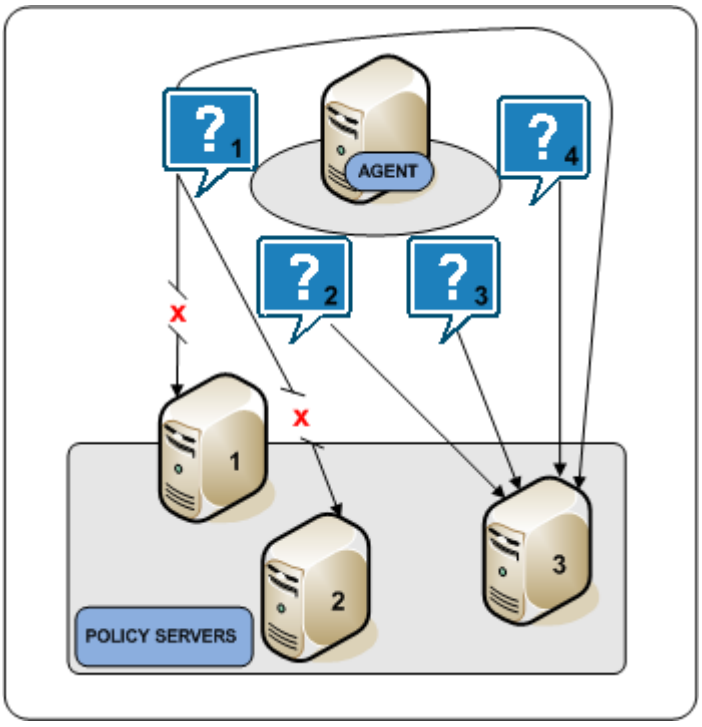
Failover is the default HCO setting. In failover mode, a Web Agent delivers all requests to the first Policy Server that the HCO lists and proceeds as follows:

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

Note: For more information about configuring an HCO with multiple Policy Servers, see the *Policy Server Configuration Guide*.

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

The following diagram illustrates the Web Agent failover process:



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

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 Web Agent requests

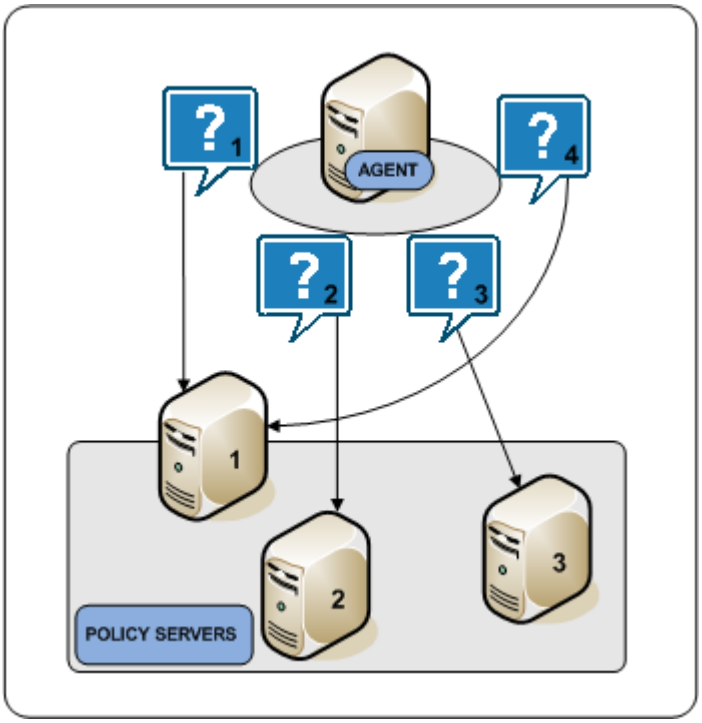
Note: For more information about configuring an HCO for round robin load balancing, see the *Policy Server Configuration Guide*.

In round robin mode, a Web Agent distributes requests across all Policy Servers that the HCO lists. A Web 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 Web Agent has sent requests to all available Policy Servers. After sending requests to all available Policy Servers, the Web Agent returns to the first Policy Server and the cycle begins again.

If a Policy Server does not respond, the Web Agent redirects the request to the next Policy Server that the HCO lists. If the unresponsive Policy Server recovers, which the Web 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:



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

Policy Server Clusters

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

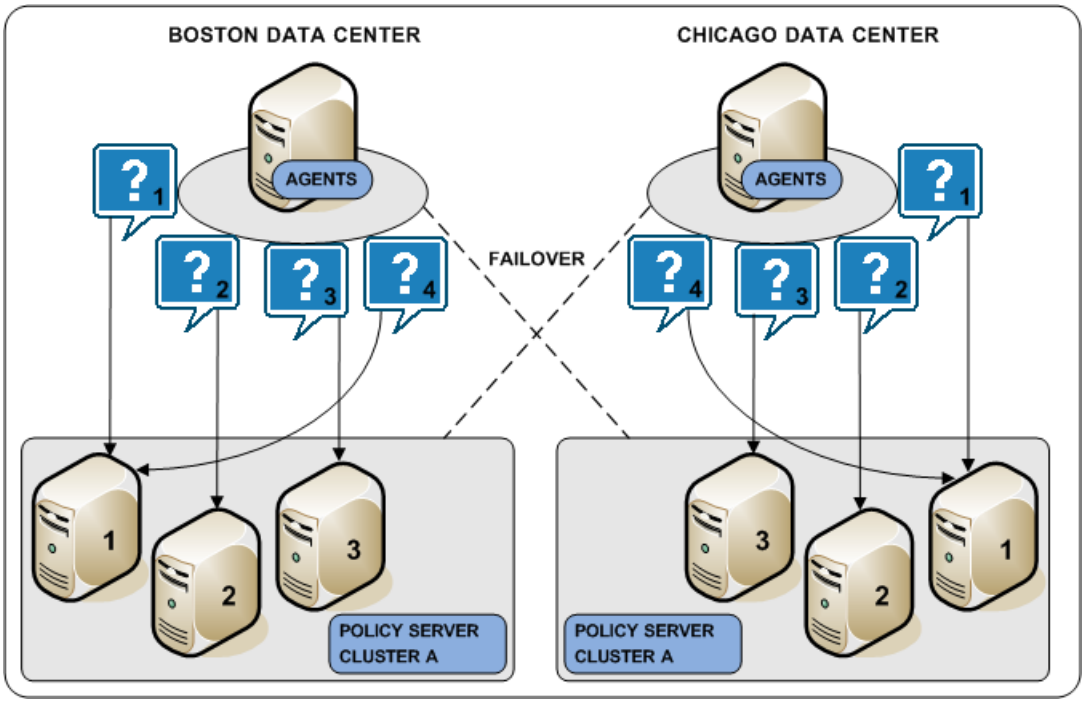
- Round robin load balancing is not the most efficient distribution method in a heterogeneous environment where computing capacity can differ. Each Policy Server receives the same number of requests, regardless of capacity.
- Round robin load balancing to Policy Servers that are located in different geographical locations can degrade performance. Sending Web 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 Web 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 Web Agents with distinct Policy Server clusters avoids the network overhead involved with load balancing across geographically separate regions. Network overhead is only incurred if Web Agents failover to another Policy Server cluster.
- A cluster can failover to another cluster based on a Policy Server failover threshold.
- Web Agents dynamically distribute requests across all Policy Servers based on response time, instead of distributing requests evenly.

Note: For more information about configuring a Policy Server cluster, see the *Policy Server Administration Guide*.

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.



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

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

Note: For more information about configuring user store connections, see the *Policy Server Configuration Guide*.

More information:

User Store (see page 17)

Failover

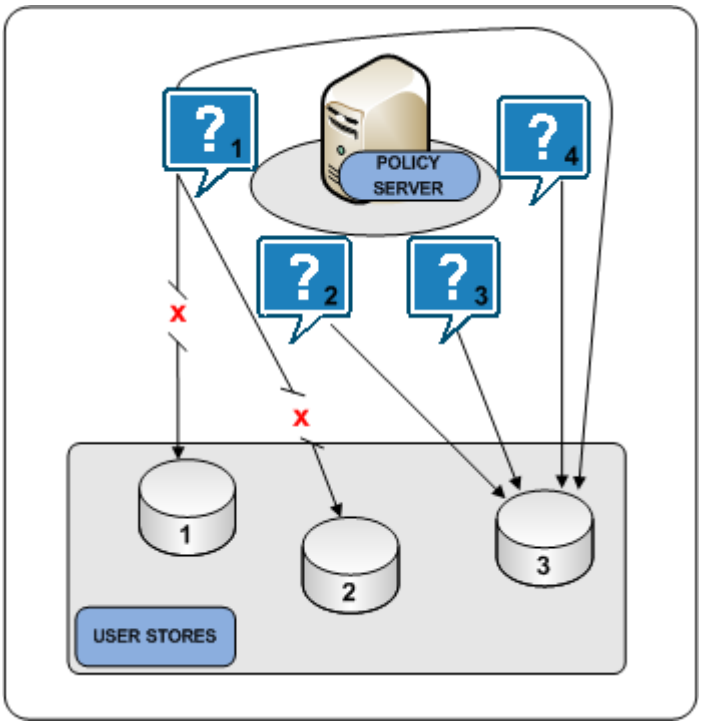
Failover is an optional setting in the SiteMinder user store object. In failover mode, a Policy Server distributes all requests to the primary user store and proceeds as follows:

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 SiteMinder 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 SiteMinder user store object lists.

Note: For more information about configuring user store failover, see the *Policy Server Configuration Guide*.

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:



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

Round Robin Load Balancing

Round robin load balancing is an optional SiteMinder 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:

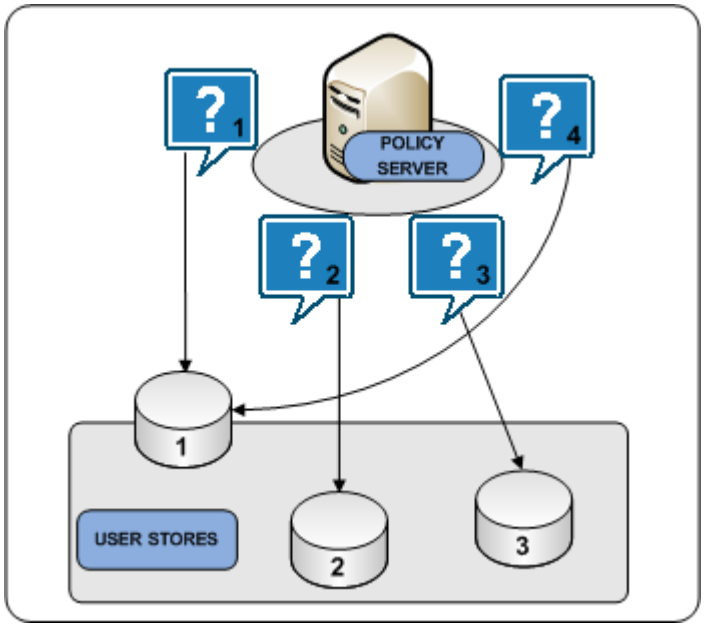
- For more information about configuring load balancing between LDAP user stores, see the Policy Server Configuration Guide.
- The Administrative UI does not include settings to configure round robin load balancing between ODBC user stores. However, the Policy Server installation includes:
  - The SiteMinder 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 SiteMinder 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 SiteMinder 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.

Note: Configure load balancing with failover to add the benefit of redundancy in the event of a user store failure. For more information about configuring load balancing and failover, see the Policy Server Configuration Guide.

The following diagram illustrates the user store round robin process:



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

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

More information:

Policy Store (see page 17)

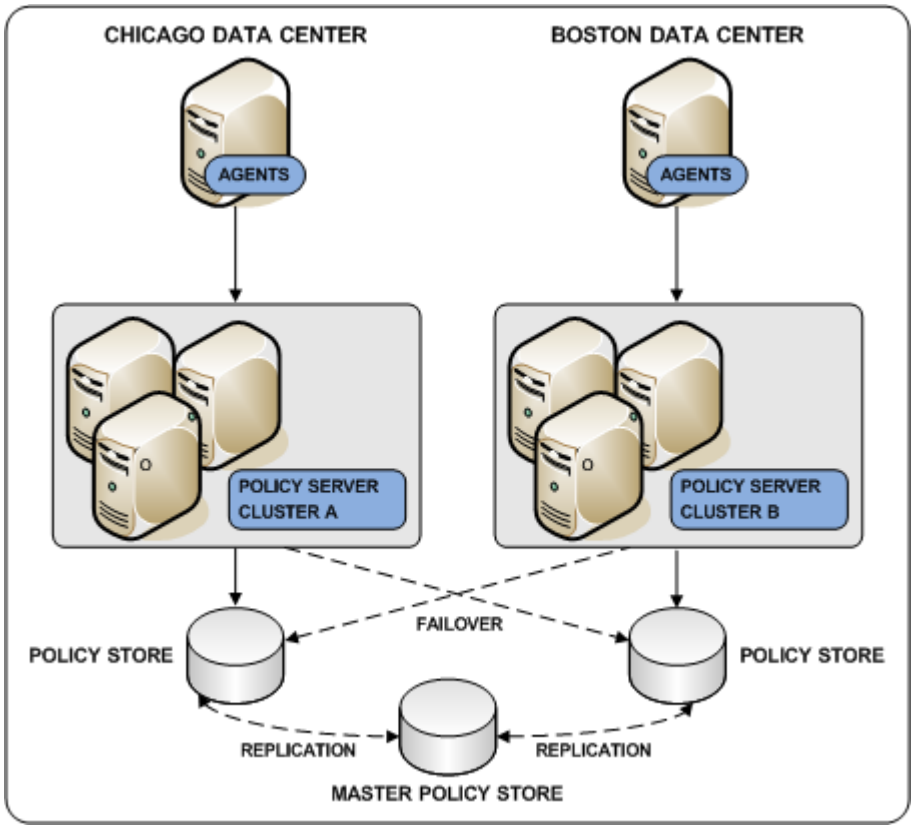
Master Policy Store

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.

Note: For more information about configuring replication, see your vendor–specific documentation. For more information about configuring policy store failover, see the *Policy Server Administration Guide*.

The following diagram illustrates a single master policy store environment:



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

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.

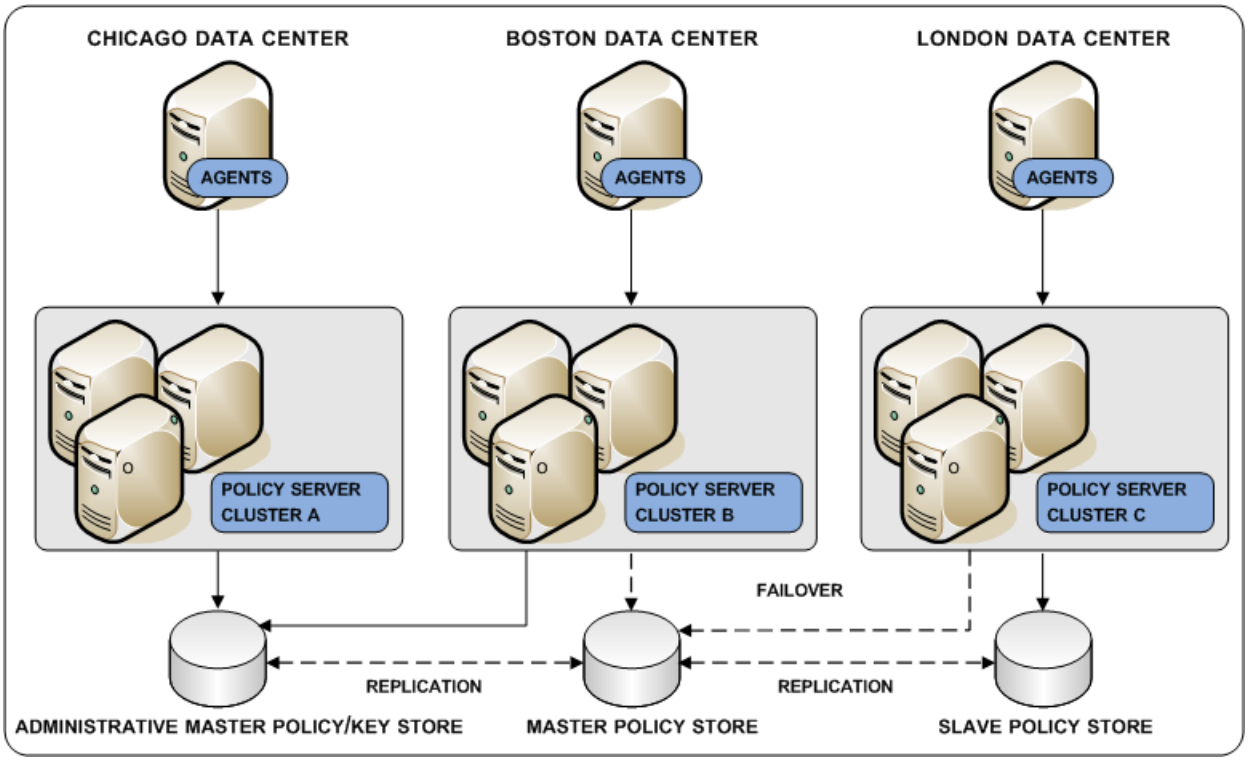
  This master does not need to be the same as the master used for Administration. However, we recommend that you use the same master store for both keys and administration. In this configuration, all key store nodes should point to the master rather than a replica.

  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 SiteMinder Support for assistance with other configurations.

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



More information:

Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

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.

SiteMinder 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.

  Note: For more information about configuring an audit store, see the *Policy Server Installation Guide*. For more information about configuring failover, see the *Policy Server Administration Guide*.

  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 Web 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. For more information about configuring failover, see the *Policy Server Administration Guide*.

- 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.

  Note: For more information about Policy Server logging and the smauditimport tool, see the *Policy Server Administration Guide*.

More information:

SiteMinder Audit Database (see page 20)

Policy Server to Session Store Communication

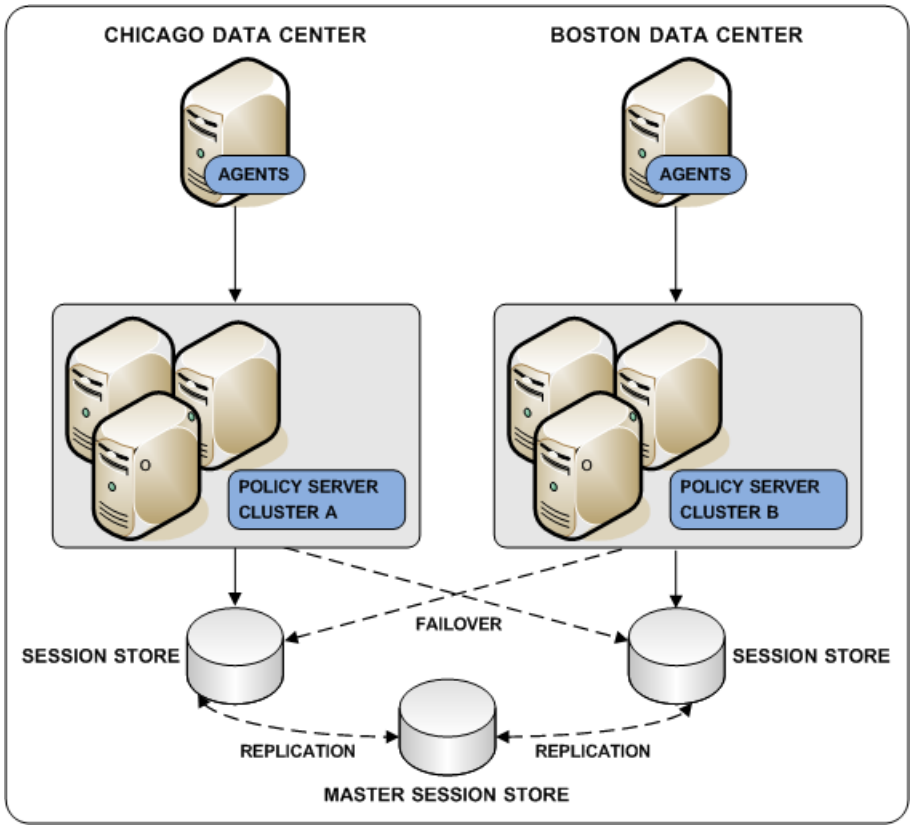
If you deploy a session store, all Policy Servers in the environment must use the same session store database.

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.

Note: For more information about configuring replication, see your vendor–specific documentation. For more information about configuring session store failover, see the *Policy Server Administration Guide*.

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



More information:

Session Store (see page 21)
Multiple Components for Operational Continuity (see page 56)
Clustered Components for Scale (see page 57)

Chapter 3: Capacity Planning

This section contains the following topics:

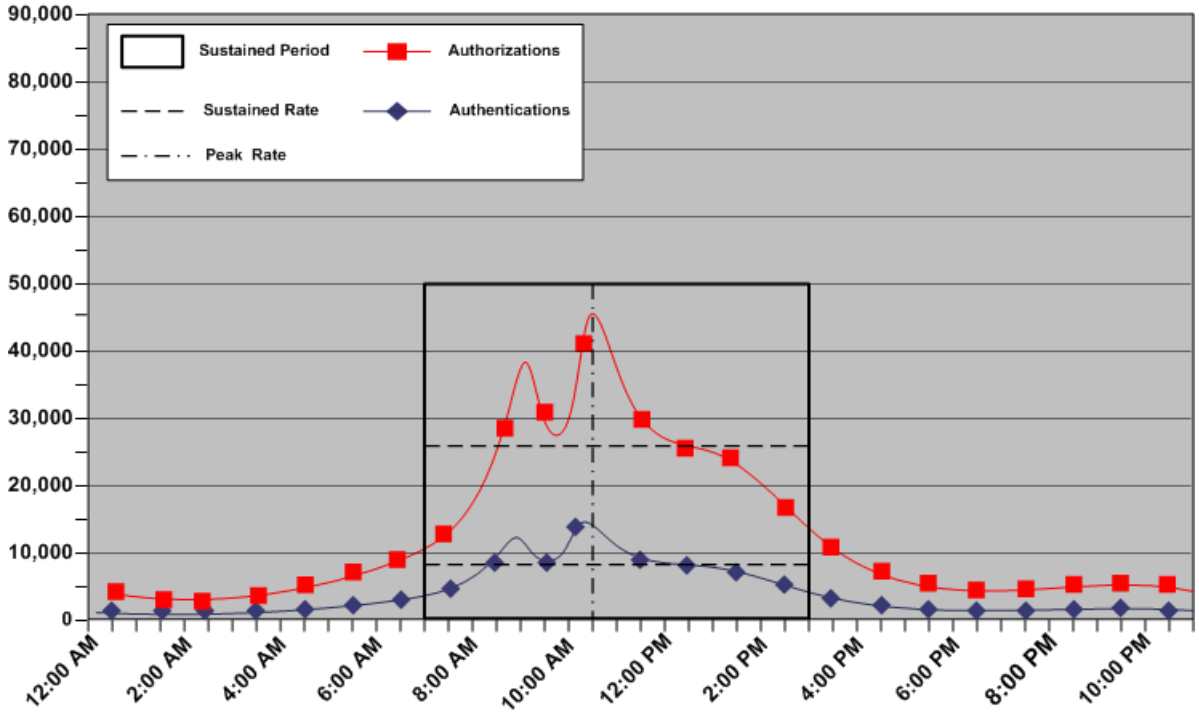
Capacity Planning Introduced

Planning a SiteMinder 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 SiteMinder must handle per application. The following general factors influence SiteMinder 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 SiteMinder 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 SiteMinder must service the highest number of authentication and authorization requests.

  Note: Although a number of other factors can influence SiteMinder 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 SiteMinder 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.

More information:

Performance Tuning Introduced (see page 111)

Use Case: Capacity Planning

The purpose of the following use case is to illustrate how a fictitious organization approaches capacity planning by modeling the usage of their application. The use case is referenced throughout this chapter for examples.

The company is planning to deploy SiteMinder. The company has 100,000 users in a single user store. Password Services is enabled for this store.

Some users log into the portal application once a day, while other users login as much as three times per day.

How to 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.

Complete the following steps to estimate the sustained authentication rate for an application:

1. Estimate daily authentications.
2. Estimate the sustained authentication rate.

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, SiteMinder 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
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?

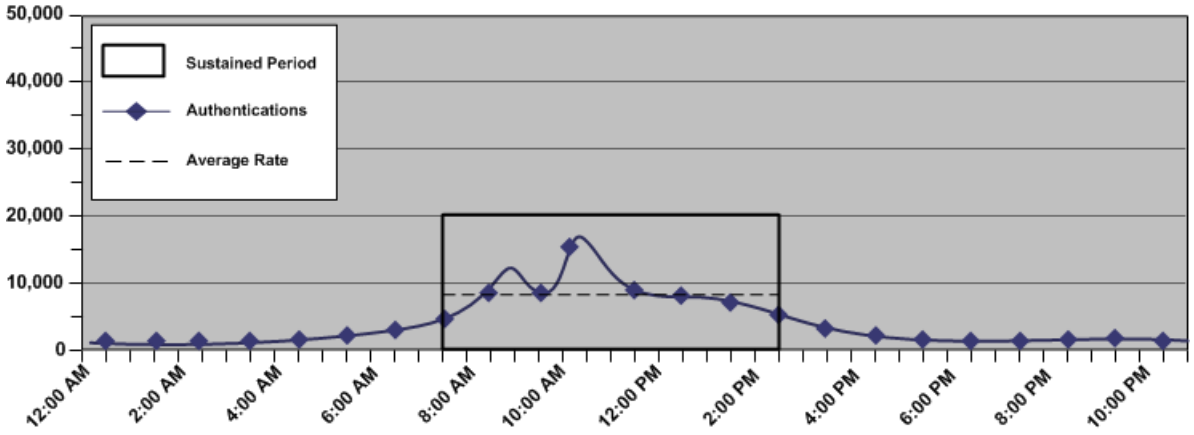
The sustained authentication rate is based on the authentication load. Specifically, when and at what rate the authentications occur. The chance that the authentication load is uniformly spread across your business day is unlikely. Rather, the rate at which requests occur fluctuates, remaining between 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 SiteMinder 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 SiteMinder 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

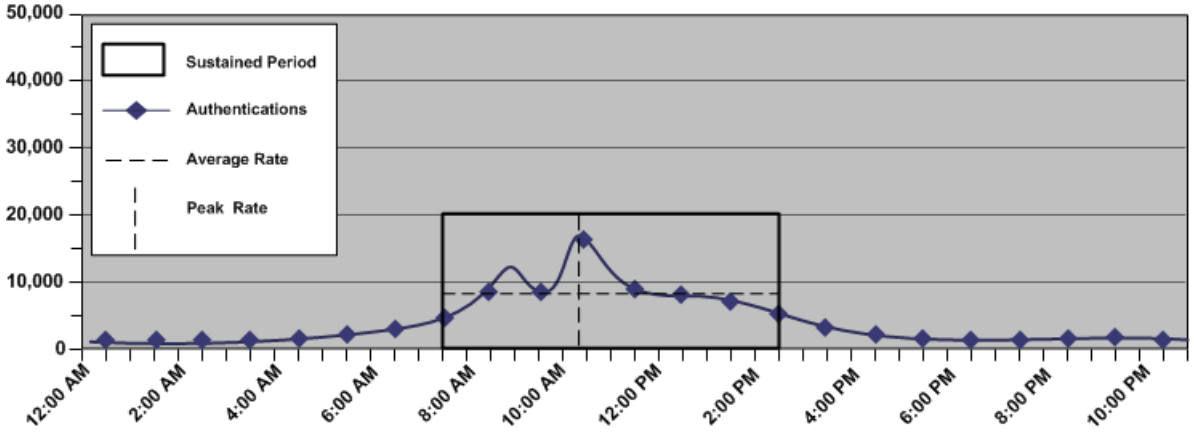
What is the peak authentication rate for the application?

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 SiteMinder 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.

More information:

Increase the Amount of Available Sockets for the Web Agent (see page 120)

How to 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.

Complete the following steps to estimate the peak authorization rate for an application:

1. Estimate daily authorizations.
2. Estimate the sustained authorization rate.

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. For the purpose of this guide, each of the following examples assume that a page visit generates one GET/POST. For more information about configuring a Web Agent to allow access to specific resources types without checking policies, see the *Web Agent Configuration Guide*.

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 (see page 77), 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.

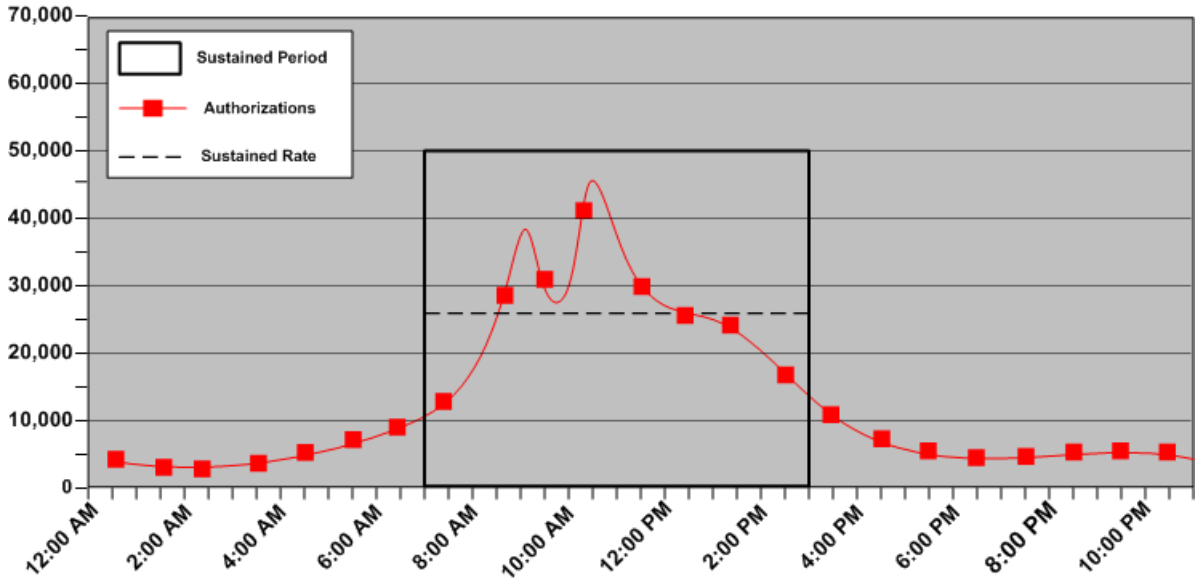
When estimating a sustained authorization rate, we recommend that you use the daily authorization load to determine:

- 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 SiteMinder 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 SiteMinder must service during the period of sustained activity.

Example: Estimate a Sustained Authorization Rate

As detailed in Estimate Daily Authorizations (see page 83), 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

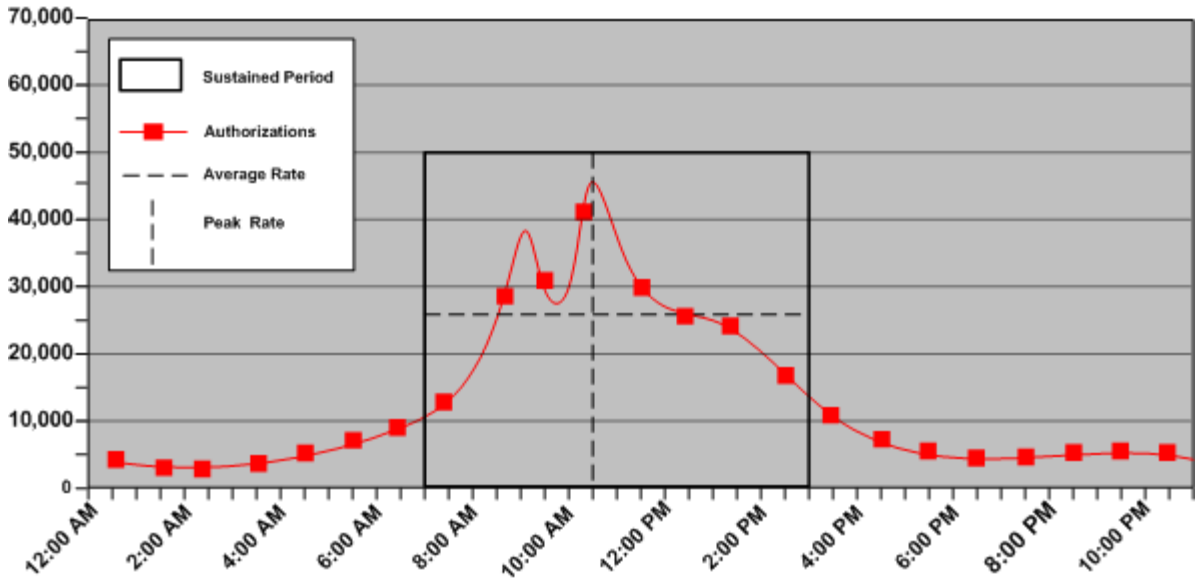
What is the peak authentication rate for the application?

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 SiteMinder 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 (see page 83), 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.

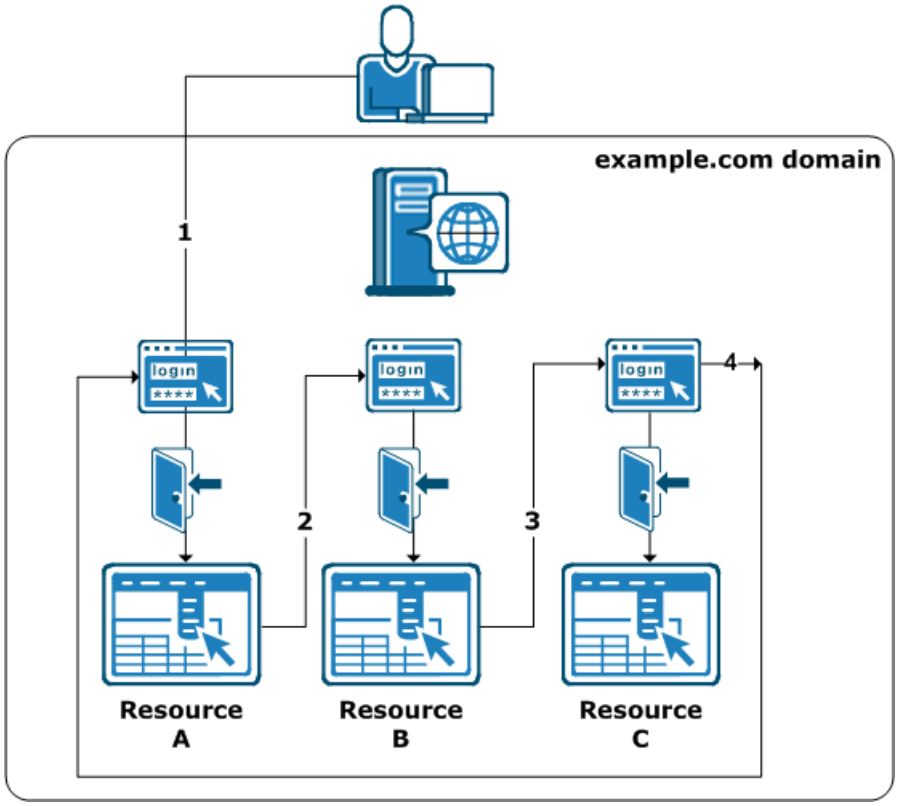
Chapter 4: Configuration Considerations

This section contains the following topics:

Security Zones

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

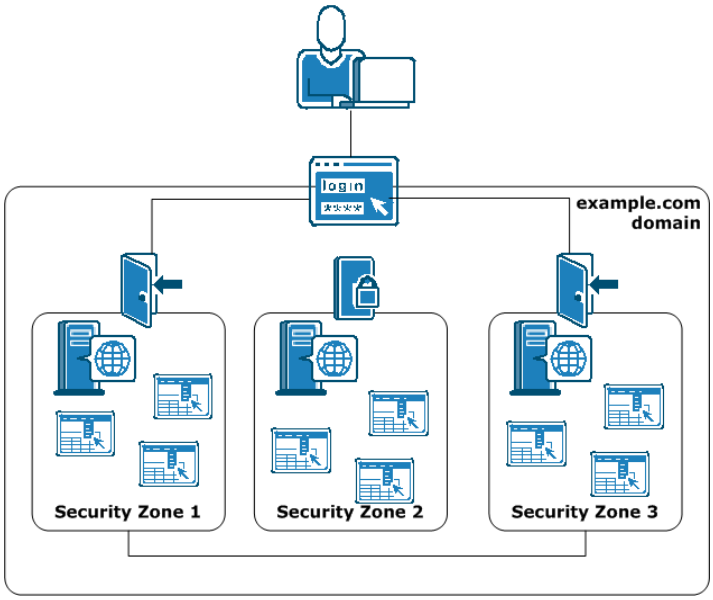
Without Security Zones, users could possibly be challenged each time they access a protected resource in the same cookie domain; even if they have previously been authenticated by SiteMinder for another resource in the cookie domain. The following illustration shows an example:



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 SiteMinder 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 SiteMinder instance could possibly overwrite the cookies of another SiteMinder 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:



Note: For more information, see the *Web Agent Configuration Guide*.

Multiple Data Centers

SiteMinder treats a global deployment the same as multiple data centers in the same continent. As such, factors outside of SiteMinder 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 SiteMinder performance:
  - Web Agents
  - Policy Servers
  - User stores

    Note: If a SiteMinder 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 SiteMinder data center:

- SiteMinder Password Services attempts to perform an LDAP write to the user account on every authentication.

  Note: For more information about Password Services, see the *Policy Server Configuration Guide*.

- SiteMinder 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 SiteMinder 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

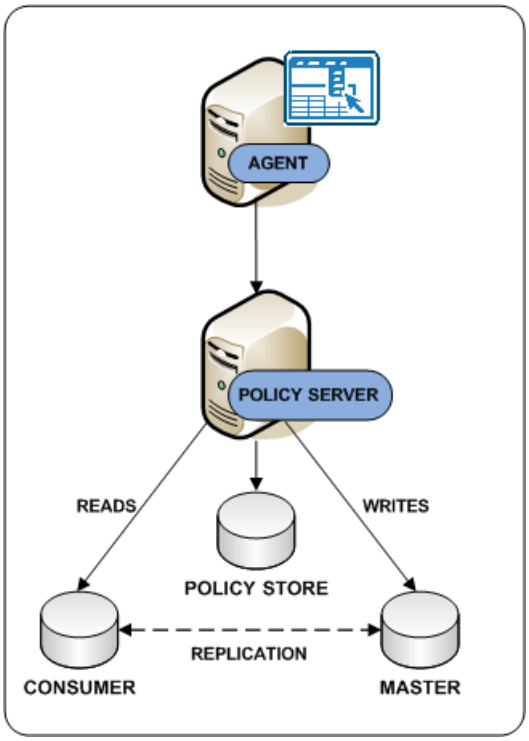
The simplest deployment includes all required SiteMinder components in a single data center.

The following diagram illustrates:

- All applications in a single data center.
- A Policy Server writing to a master user store. SiteMinder 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*.

- SiteMinder reading a consumer user store.



Consider the following:

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

More information:

SiteMinder Components (see page 13)
Multiple Components for Operational Continuity (see page 56)
Redundancy and High Availability (see page 60)

All Components in Multiple Data Centers

You extend the SiteMinder environment by deploying multiple data centers. The following factors can influence your decision to implement multiple data centers:

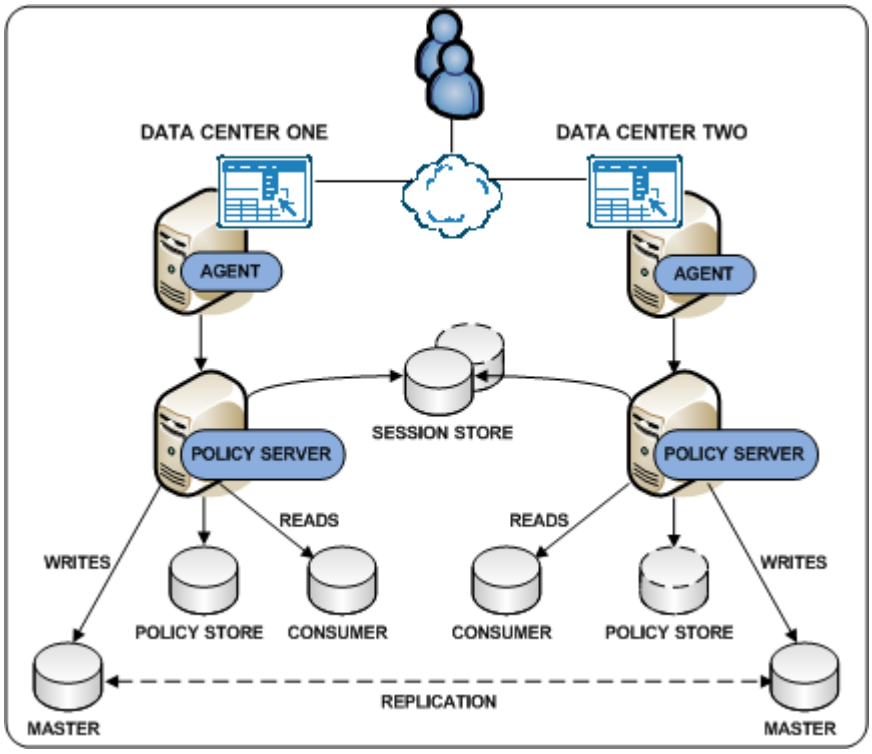
- The network infrastructure
- The location of applications
- The location of users

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 detailed by the dotted line.

  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 68).
- Each data center using its own master/consumer user stores (see page 95).

  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.



More information:

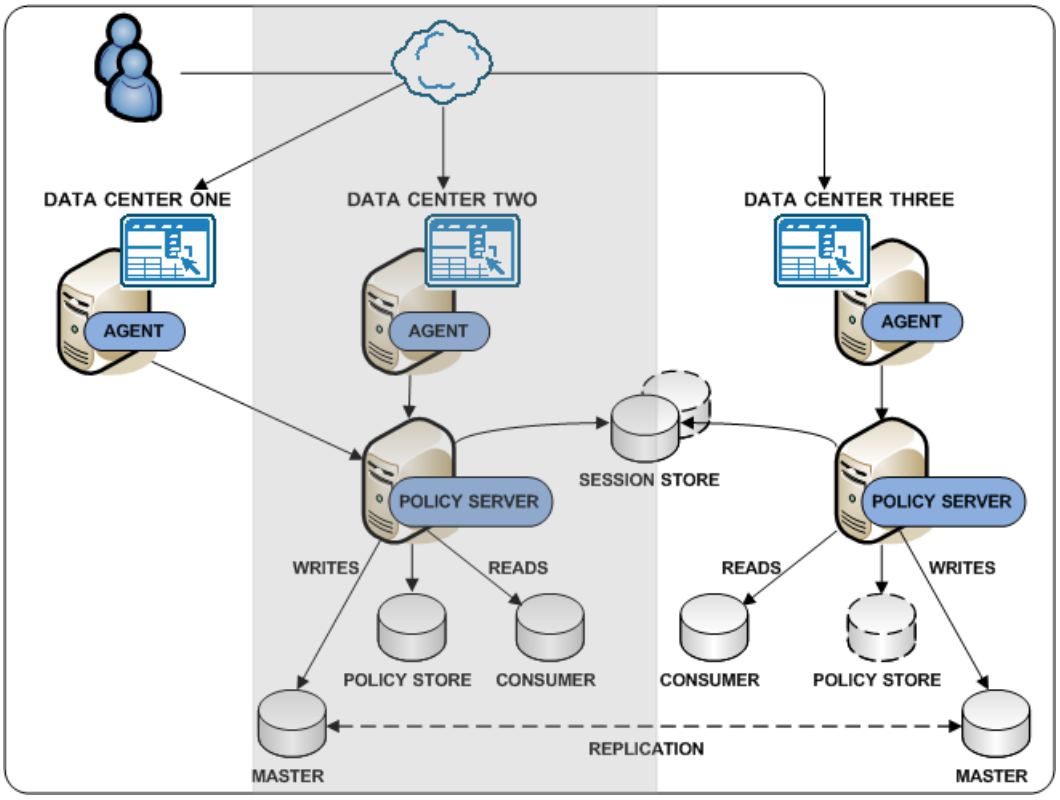
Policy Server to Policy Store Communication (see page 68)

Web 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 SiteMinder Web Agent. The Web 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 69).
  - using their own master/consumer user stores (see page 95).
  - using a centralized replicated session store to enable single sign–on between all applications.



More information:

Policy Server to Policy Store Communication (see page 68)

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:

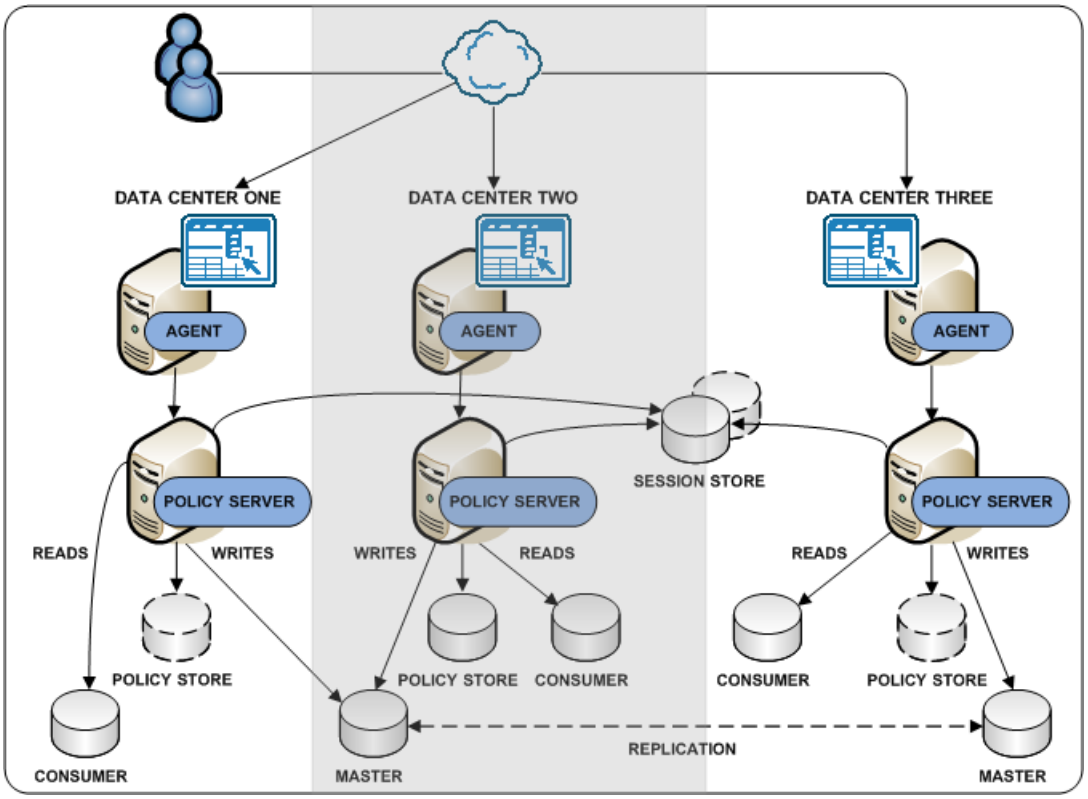
- Applications in multiple data centers.
- Data center 1 only containing a Web Agent and Policy Server. The Policy Server only communicates across the wide area network to perform LDAP writes to the master user store in data center 2.

  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 69).
  - 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 95).

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



More information:

Policy Server to Policy Store Communication (see page 68)

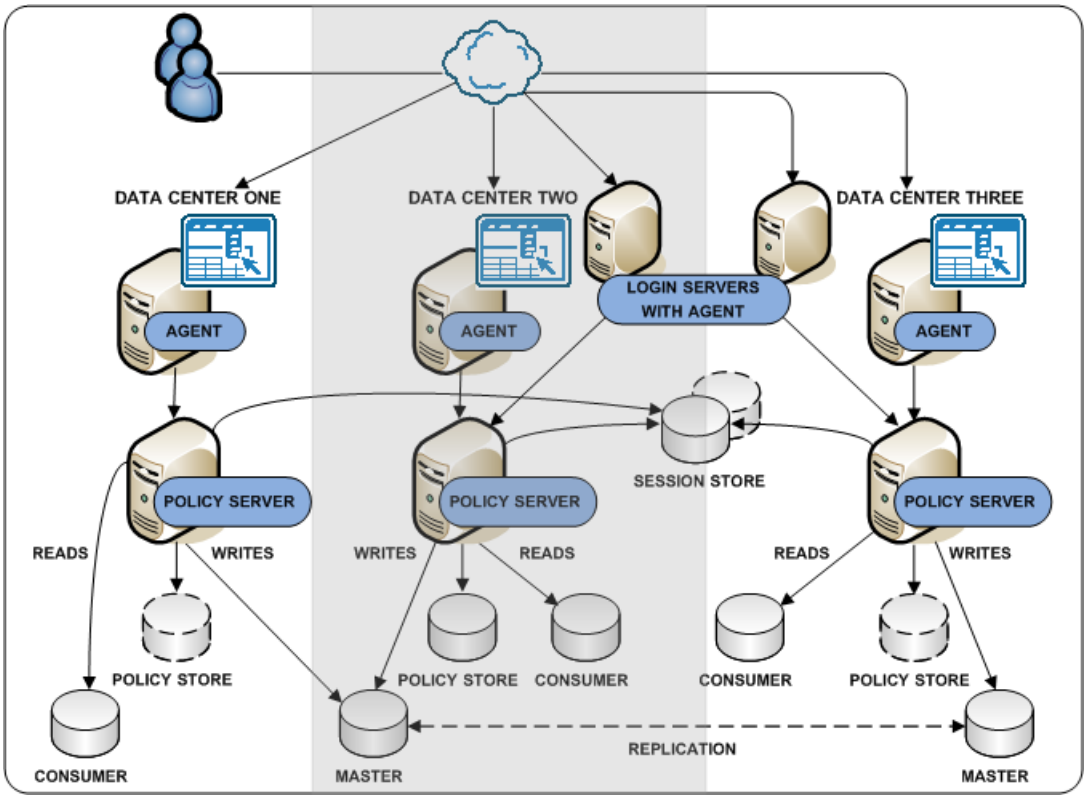
Login Server Controlling User Store Writes

The location of LDAP writable masters can constrain a SiteMinder 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 99).
  - The remaining data centers including all components (see page 96).

- A login server in data center two and data center three.



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.

   Note: For more information about authentication schemes, see the *Policy Server Configuration Guide*.

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

3. The Policy Server creates a SiteMinder session ticket and passes it back to the original protected URL.

   Note: For more information about user sessions, see the *Policy Server Configuration Guide*.

4. A Web Agent places the SiteMinder session ticket into a cookie, which it uses to handle subsequent authentication and authorization requests in the data center until the user requests another resource that requires additional credentials or the session expires.

Authentication and a Centralized Login Server

A SiteMinder 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 SiteMinder 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.

Note: For more information about authentication schemes, see the *Policy Server Configuration Guide*.

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 SiteMinder 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 SiteMinder 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.

By identifying applications that share the same authentication requirements, three login pages replace the need for eight, as detailed by the following table:

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 SiteMinder OnAuthAttempt and OnAuthReject responses to redirect users.

  Note: For more information about responses, see the *Policy Server Configuration Guide*.

  - 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.

    Note: For more information about the login FCC as it applies to forms–based authentication, see the *Policy Server Configuration Guide*. For more information about configuring the login FCC with a Web Agent and how the FCC process requests, see the *Web Agent Configuration Guide*.
- 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.

  Note: For more information about configuring a logoff page, see the *Web Agent Configuration Guide*.

Login Page Use Cases

The purpose of the following use cases is to get you thinking about configuring SiteMinder 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, SiteMinder directs users to a stand–alone login page when they request a protected resource. Specifically:

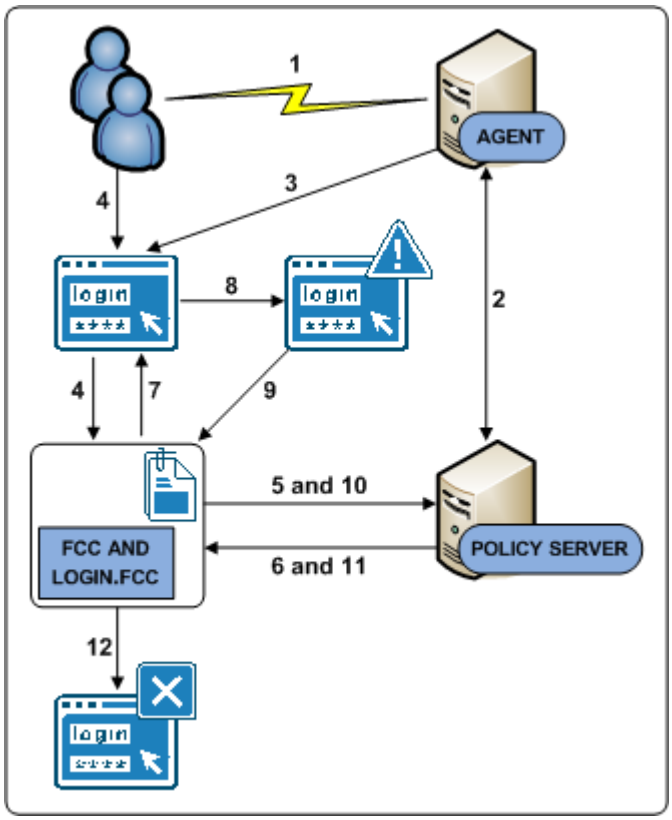
- A dynamic login page (login.asp) is deployed to the Web Agent host system.
- The dynamic login page is coded to:
  - Post to a login FCC file (login.fcc).
  - Display an error message when the SMTRYNO cookie is present in the web browser of the user.

    Note: For more information about the SMTRYNO cookie, see the *Web Agent Configuration Guide*.
- 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.

  Note: For more information about configuring an FCC file with @directives, see the *Policy Server Configuration Guide*.
- A SiteMinder administrator has configured a form–based authentication scheme named Auth1. The target of Auth1 is login.asp.

  Note: For more information about configuring authentication schemes, see the *Policy Server Configuration Guide*.

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



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:

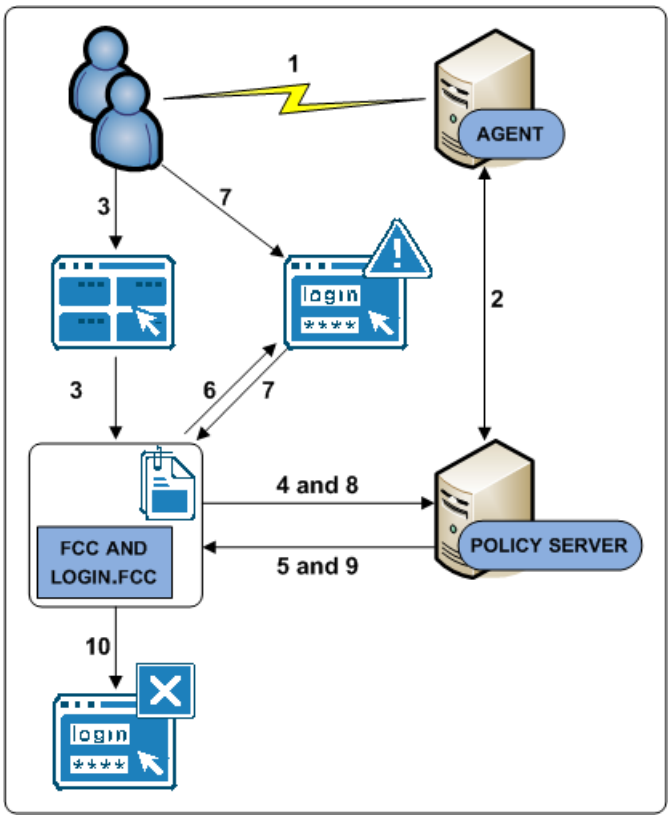
- A web portal home page (portal.asp) includes an embedded form that prompts users for credentials. The home page:
  - Contains a target variable that points to the protected resource.
  - Posts to a login FCC file (login.fcc).
- A stand-alone login page (login.asp) is deployed to the Web Agent host 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.

    Note: For more information about the SMTRYNO cookie, see the *Web Agent Configuration Guide*.
- 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.

  Note: For more information about configuring an FCC file with @directives, see the *Policy Server Configuration Guide*.
- A SiteMinder administrator has configured a form–based authentication scheme named Auth1. The target of Auth1 is login.asp.

  Note: For more information about configuring authentication schemes, see the *Policy Server Configuration Guide*.

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



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.

Chapter 5: Performance Tuning

This section contains the following topics:

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 SiteMinder components. SiteMinder performance tuning is the iterative process of increasing throughput and reducing latency by:

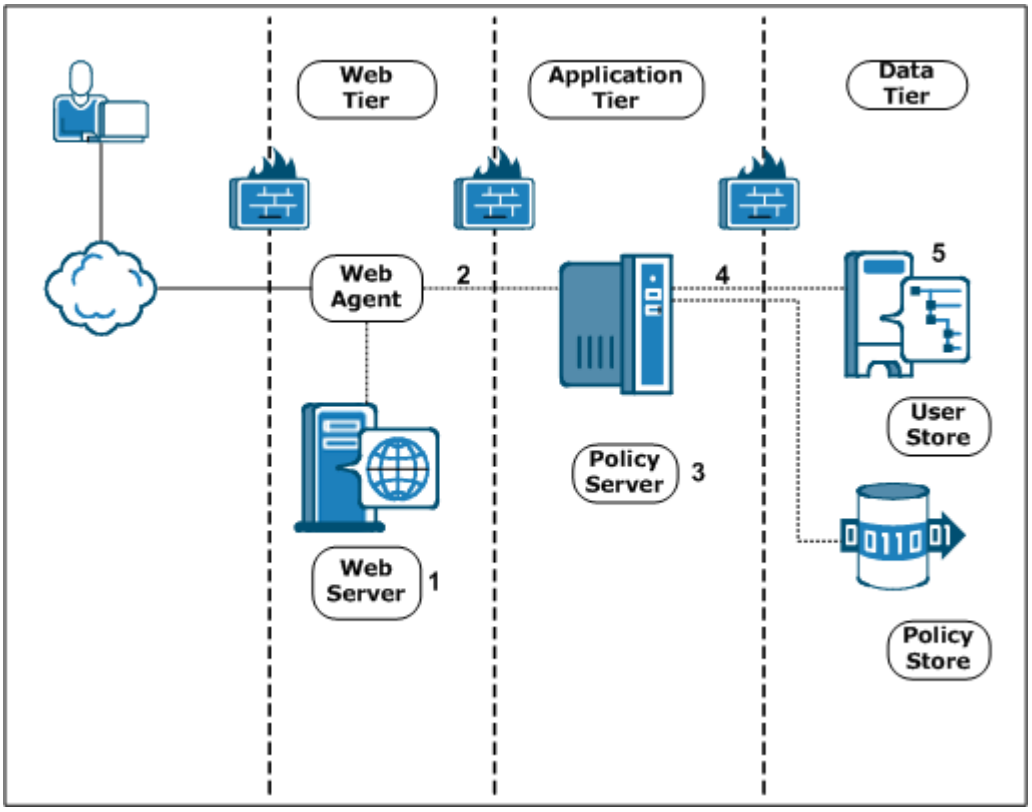
- Understanding where and when these transactions occur
- Identifying the SiteMinder settings and features that affect performance
- Using third-party and SiteMinder tools to measure performance and identify infrastructure bottlenecks

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

Note: SiteMinder is middleware and is not deployed independently. The following sections focus on tuning SiteMinder 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 SiteMinder Agents, Policy Servers, or the SiteMinder 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 servers deployed in your environment can affect how a SiteMinder Web Agent and Policy Server communicate. For more information, see Web Server Performance. (see page 114)
2. The number of available sockets can affect the efficiency in which a Web Agent and Policy Server communicate. For more information, see Web Tier Socket Usage (see page 119) and Reduce Traffic between your Web Agent and the Policy Server (see page 122).
3. SiteMinder policy design can affect the efficiency in which the Policy Server services authentication and authorization requests. For more information, see SiteMinder Policy Design and Performance (see page 131).

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 SiteMinder performance is determining whether your user directories can handle this workload during sustained and peak periods of operation. For more information, see User Store Capacity Planning (see page 150).
5. The user directory itself can affect SiteMinder performance. For more information, see Data Tier Guidelines (see page 147).

Web Tier Performance

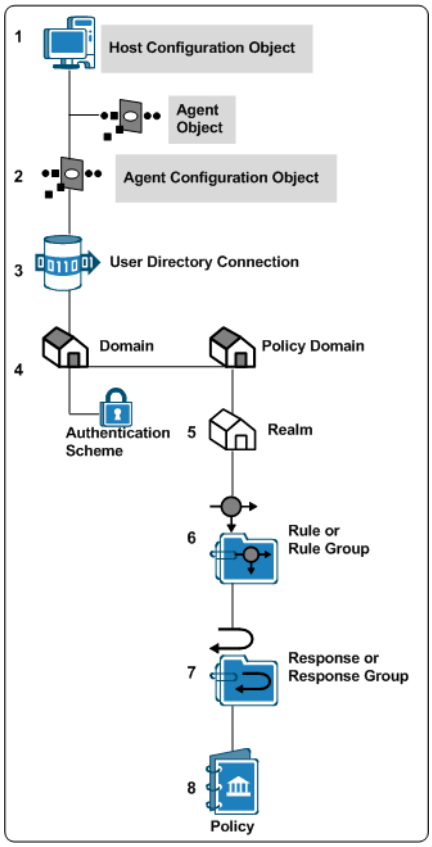
When a SiteMinder Web Agent intercepts a request sent to a Web Server, the Web Agent makes the following calls to the SiteMinder Policy Server:

- isProtected
- isAuthenticated
- isAuthorized

Each of the previous calls generates traffic between the Web 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 a Web Agent to use for Policy Server connections.
- Use the Web Agent caches to reduce the number of calls a Web Agent makes to the Policy Server.

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



Web Server Performance

The Web Agent is installed on a web server. The performance of your web server correlates to the performance of the web tier in SiteMinder. The following items affect how your web server performs with SiteMinder:

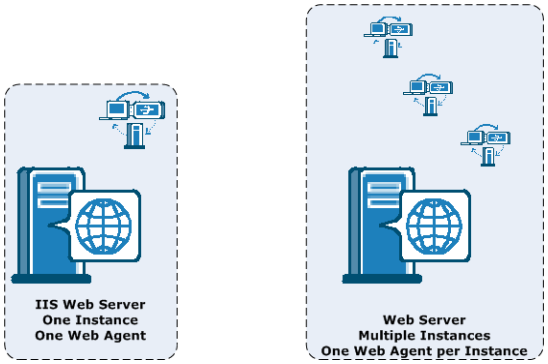
- The processor speed of your web server
- The amount of memory in your web server

Web Servers, Web Agents, and Web Server Processes

Each Web 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 Web Agents equals the number of IIS web servers.

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

The following illustration shows an example:



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

Pre-fork mode

Creates child processes to handle additional requests.

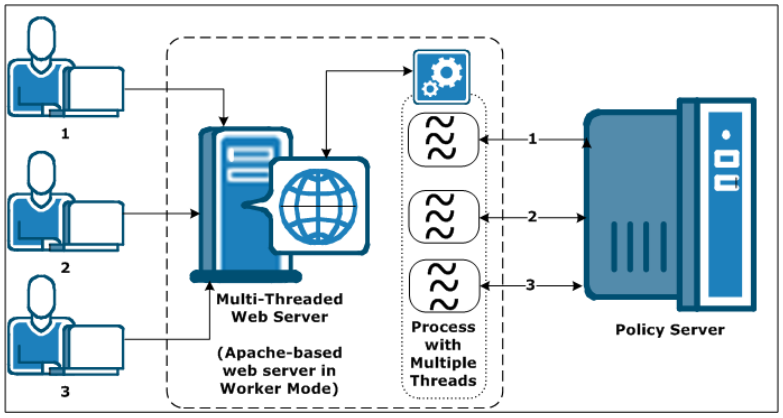
Worker mode

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 SiteMinder 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:



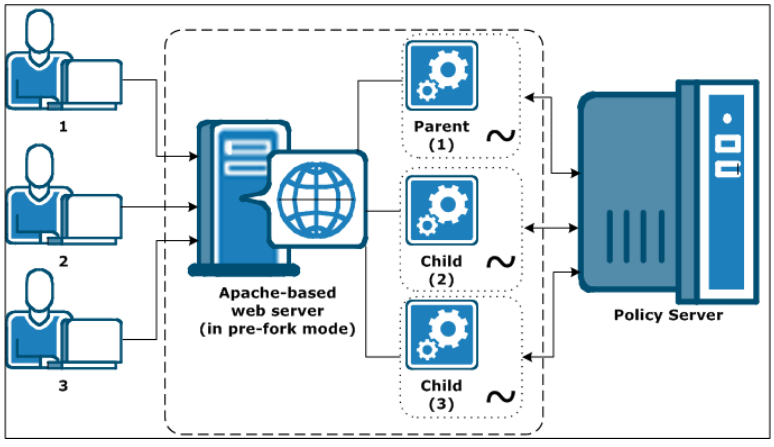
More information:

Failover and Load Balancing with Multi-Threaded Web Servers (see page 129)

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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder Policy Servers receiving the requests can handle the maximum number of connections that the related web servers could open.

More information:

Failover and Load Balancing with Multi-Process Web Servers (see page 130)

Web Agent Performance

The following factors influence SiteMinder Web Agent performance:

- Web server CPU and available memory.
- Policy Sever latency (how quickly the Policy Server responds to Web 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 server during peak periods can help you determine the ideal number of web servers for your SiteMinder environment.

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

- Complete a capacity planning effort.
- Generate a SiteMinder Activity report for each Web Agent in your environment.
- Generate a performance report for your web server.

  Note: For more information, see the documentation provided by your web server vendor.

More information:

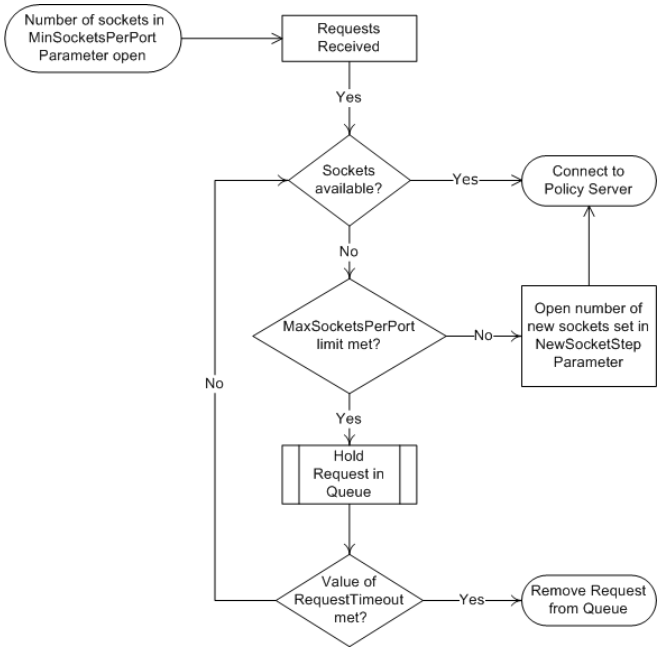
Estimate a Peak Authentication Rate (see page 81)
Estimate a Peak Authorization Rate (see page 88)

Web Tier Socket Usage

When a Web 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 Web 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 following illustration describes this process:



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 Web 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 Web 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.

Note: For more information, see the *SiteMinder Policy Server Configuration Guide*.

Increase the Amount of Available Sockets for the Web Agent

If your capacity planning estimates reveal that the number of user requests per Web 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 Web Agent processes in your SiteMinder 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 SiteMinder Policy Server, only one socket is required.

More information:

Estimate a Peak Authentication Rate (see page 81)

Increase NewSocketStep Setting

When the Web 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 Web 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 Web 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 Web Agent from spending too much time creating sockets for requests as the load on the web server increases.

We recommend experimenting with different settings until you find what works best in your SiteMinder 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 SiteMinder Policy Server, only one socket is required.

More information:

Estimate a Peak Authentication Rate (see page 81)

Minimum Sockets per Port Setting

When a Web 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 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 Web 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 SiteMinder Policy Server, only one socket is required.

Reduce Traffic between Your Web Agents and the Policy Server

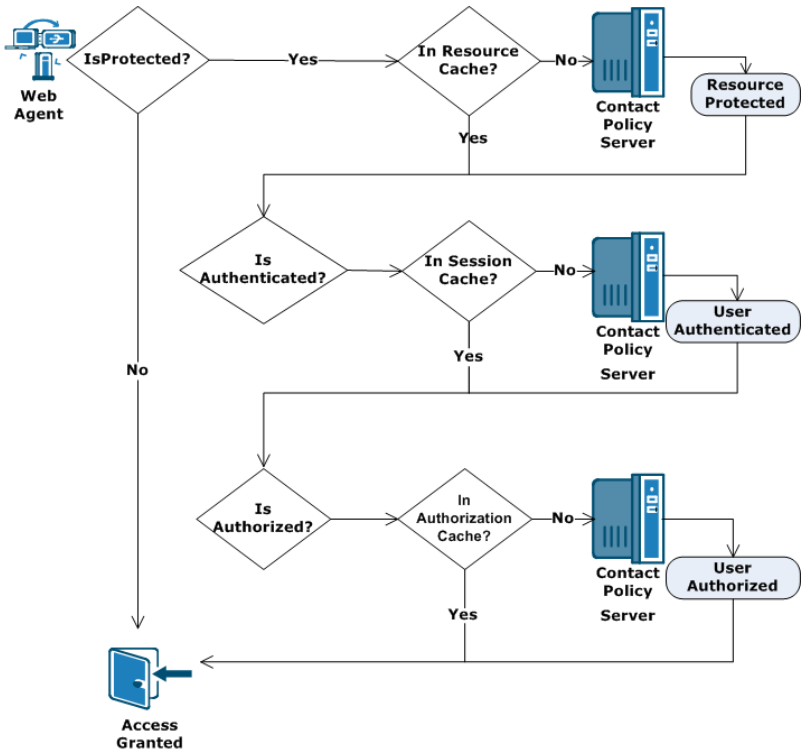
The Web Agent has multiple caches and configuration parameters that you can use together to reduce the amount of traffic between your Web Agents and Policy Servers. Generally, these settings are most efficient in SiteMinder environments where the policies and URLs usually remain static.

How Web Agent Caches Work

The Web Agent searches the following caches for the information it needs before contacting the SiteMinder 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 Web 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 Web 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 Web Agent does not make an IsProtected call to the Policy Server.

Two Web Agent configuration parameters affect the resource cache. Consider the following as you plan your SiteMinder deployment:

Resource Cache Timeout

We recommend basing the timeout interval of the Web 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 Web Agent configuration determines the timeout interval of the resource cache.

Resource Cache Size

We recommend using a resource cache that is 10 percent larger than the largest number of URIs that you expect users to request. If you are protecting an application that uses dynamic URLs (such as URLs with query strings) consider using the IgnoreQueryData parameter instead of adjusting the size of the resource cache. The value of the MaxResourceCacheSize Web Agent configuration parameter determines the size of the resource cache.

Note: For more information, see the *Web Agent Configuration Guide*.

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 URL stored 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

Note: For more information, see the *Web Agent Configuration Guide*.

Session Cache (authentication)

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

The Web 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.

Authentication information is removed from the session cache and discarded.

Authorization Cache

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

The Web 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 Web 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 SiteMinder 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.

Note: For more information, see the *SiteMinder Policy Server Configuration Guide*.

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.

Note: For more information, see the *Web Agent Configuration Guide*.

More information:

How to Estimate a Sustained Authentication Rate (see page 77)

Caching and Anonymous Users

The anonymous authentication schemes offered by SiteMinder 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. SiteMinder 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 SiteMinder 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.

Note: For more information, see the *Web Agent Configuration Guide*.

Other Parameters That Affect Web Agent Performance

The following parameters also affect Web Agent performance:

- PSPollInterval
- IgnoreExt
- IgnoreURL

Policy Server Poll Interval Parameter

Web 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 Web Agent configuration parameter.

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

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

  Note: For more information, see the *Web Agent Configuration Guide*.

Important! Increasing the PSPollInterval parameter also affects how quickly the Web Agents enforce SiteMinder 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 SiteMinder 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.

Note: For more information, see the *Web Agent Configuration Guide*.

More information:

Identify the Applications to Secure (see page 31)

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 Web Agent Performance through Load Balancing

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

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 Web 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 Web 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 Web 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.

  Note: For more information, see the *SiteMinder Policy Server Configuration Guide*.

Failover and Load Balancing with Multi-Threaded Web Servers

Web Agents running on multi-threaded web servers (such as Sun Java System, IIS, or an Apache-based server in worker mode), 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 Web 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 Web 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.

More information:

Web Agent and Policy Server Interaction using Apache-based Web Server Worker Mode (see page 116)

Failover and Load Balancing with Multi-Process Web Servers

A Web Agent running on a multi-process web 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.

More information:

Web Agent and Policy Server Interaction using Apache-based Web Server Pre-Fork Mode (see page 116)

Application Tier Performance

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 SiteMinder user directories during authentication and authorization affects performance.

SiteMinder Policy Design and Performance

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

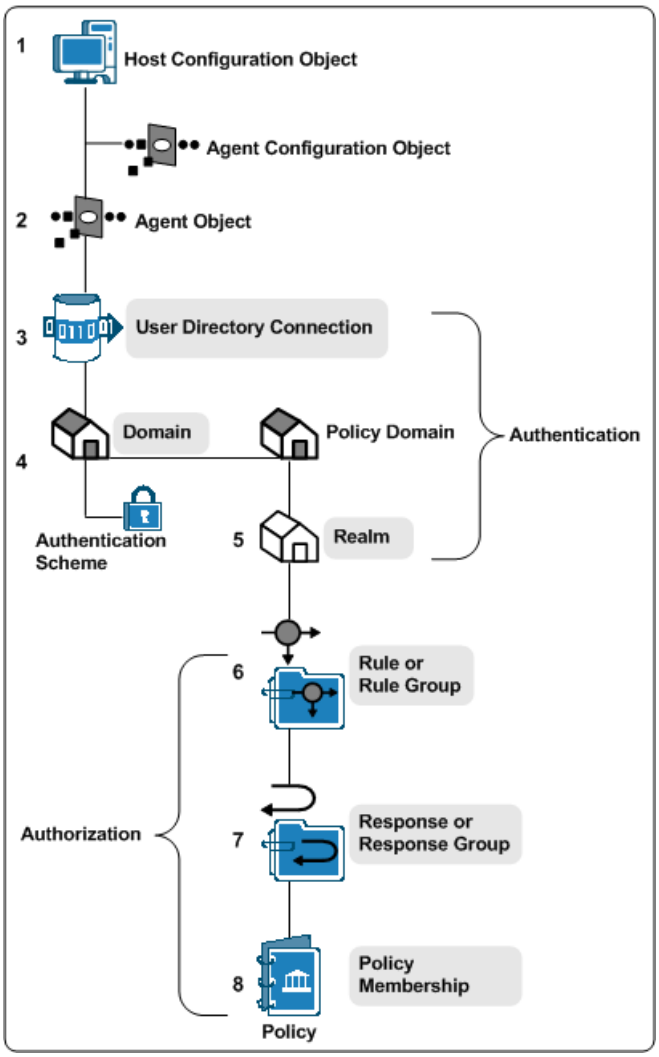
You can improve or degrade performance in the way you configure specific SiteMinder components or by choosing to enable optional features. A performance strategy includes:

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

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

SiteMinder Policy Objects and Performance Roadmap

SiteMinder requires that you configure core SiteMinder 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.

More information:

Web Tier Performance (see page 113)

Domains

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

A SiteMinder 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 SiteMinder performance during authentication.

Note: For more information about configuring domains, see the *Policy Server Configuration Guide*.

More information:

Group Resources into Domains or EPM Applications (see page 32)
Domains and Authentication Performance (see page 137)

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 SiteMinder handles user sessions. SiteMinder 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.

Note: For more information about realms, see the *Policy Server Configuration Guide*. For more information about authentication schemes, see the *Policy Server Configuration Guide*.

More information:

Group a Resources into Realms or EPM Components (see page 33)
Realms and Authentication Performance (see page 138)

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 SiteMinder performance during authorization.

Note: For more information about rules, see the *Policy Server Configuration Guide*.

More information:

Rules and Authorization Performance (see page 141)

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 a SiteMinder Agent.

Policies rules can be bound to one or more responses. The types of responses you bind to SiteMinder policy rules directly correlates to SiteMinder performance during authorization.

Note: For more information about responses, see the *Policy Server Configuration Guide*.

More information:

Responses and Authorization Performance (see page 142)

Authentication Guidelines

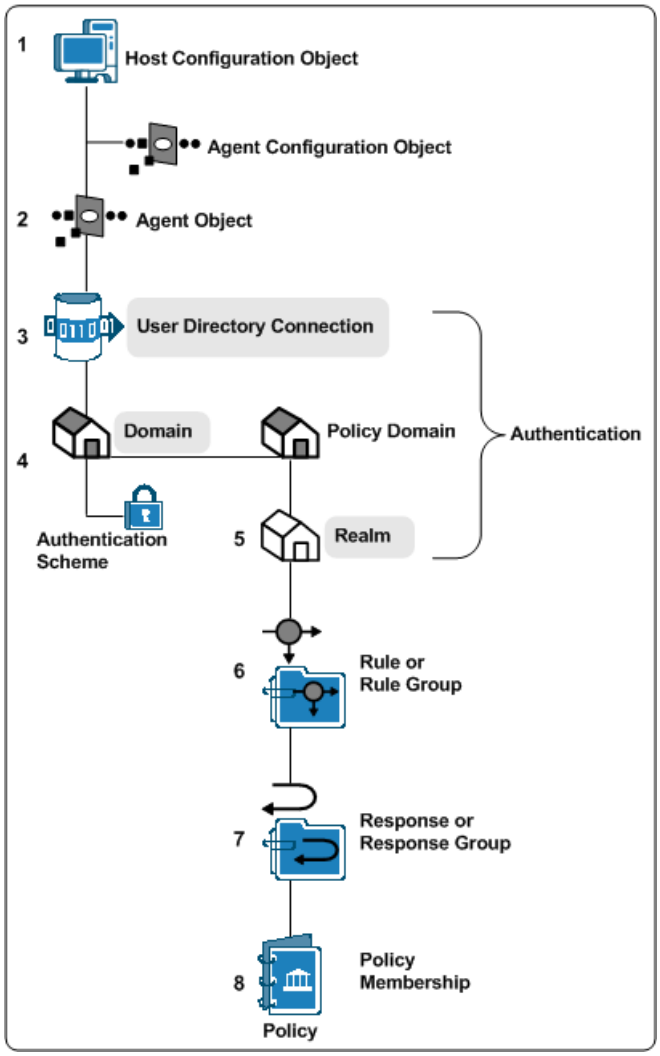
SiteMinder performance during the authentication (IsAuthenticated?) step typically correlates with:

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

SiteMinder Policy Objects and Performance Roadmap

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

SiteMinder requires that you configure core SiteMinder 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.

Note: For more information about configuring user directory connections, see the *Policy Server Configuration Guide.*

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 SiteMinder 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.

Domains and Authentication Performance

The following factors affect user authentication performance at the domain 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 SiteMinder 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 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.

  Note: For more information about configuring authentication schemes, see the *Policy Server Configuration Guide*. For more information about using credential collectors, see the *Web Agent Configuration Guide*.

- Persistent Sessions—When SiteMinder 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, SiteMinder 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 SiteMinder features require persistent sessions. You can configure a realm for Persistent Sessions. Agents protecting resources in this realm write the session ticket a SiteMinder session store, which results in additional requests to the session store for each authentication.

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

  Note: For more information about user sessions, see the *Policy Server Configuration Guide*.

- 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.

  Note: For more information about realms, see the *Policy Server Configuration Guide*.

Authorization Guidelines

SiteMinder performance during the authorization step typically correlates with:

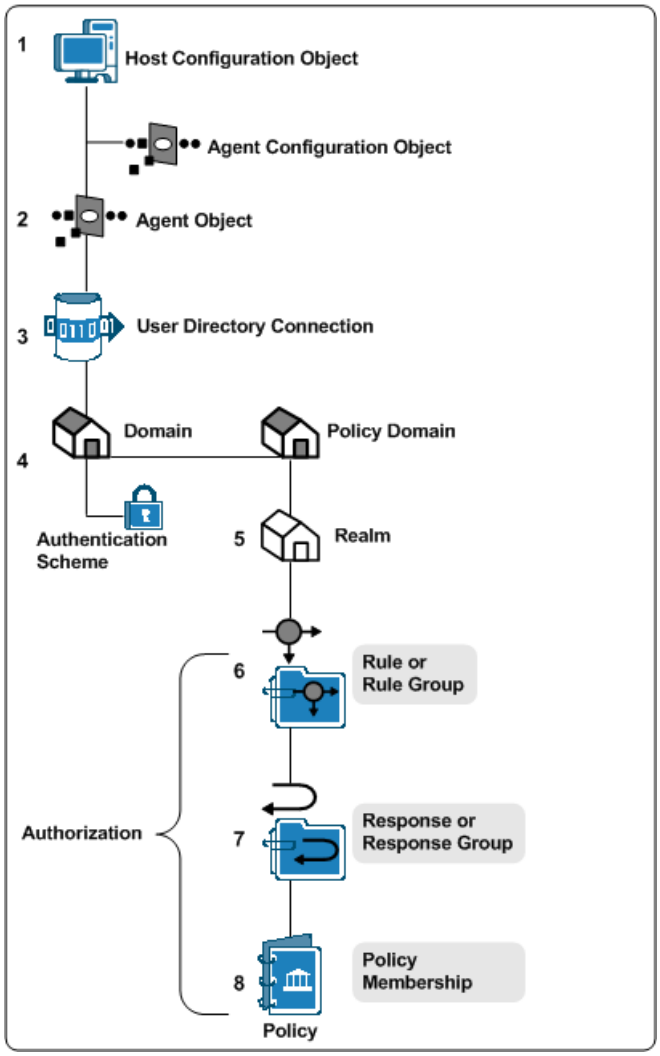
- The system resources used to service an authorization request.
- The number of reads/writes, collectively known as requests, the Policy Server makes to SiteMinder user directories to service an authorization request.

The complexity of your SiteMinder policy design affects each of these areas.

Policy Objects and Performance Roadmap

You can improve or degrade authentication performance in the way you configure specific SiteMinder policy objects or by choosing to enable optional features associated with those objects.

SiteMinder requires that you configure core SiteMinder policy objects in a specific order. The following diagram lists this order, where shaded items represents objects that affect performance during user authorization.



Rules and Authorization Performance

The following factors affect user authorization performance at the rule 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.

Note: For more information about rules, see the *Policy Server Configuration Guide*.

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 SiteMinder 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.

SiteMinder Policy Membership and Authorization Performance

Policy membership is the part of a SiteMinder policy that specifies which users apply to the policy. SiteMinder policies are stored in domains, and as a result, you use filters to apply SiteMinder 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 SiteMinder policy membership.

Note: For more information about adding users to a SiteMinder policy, see the *Policy Server Configuration Guide*.

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 SiteMinder 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 SiteMinder policy to determine that the policy applies to the user.

  Note: For more information about user sessions, see the *Policy Server Configuration Guide*.

- 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 SiteMinder 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:

  a. Resolve the group membership or search expression

  b. Search the user directory to determine if the SiteMinder policy applies to the user.

- Nested groups—Defining SiteMinder 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 SiteMinder 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.

More information:

User Authorization Cache (see page 143)

User Authorization Cache

The user authorization cache reduces the number of user directory requests to determine SiteMinder 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 SiteMinder policy membership, it must resolve the group membership and make three requests (one for each policy) to the user directory to determine that each SiteMinder 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.

More information:

SiteMinder Policy Membership and Authorization Performance (see page 142)

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 SiteMinder 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 SiteMinder agent round–robin pool, the greater the chance that the efficiency of the user authorization cache is reduced.

If a single Web 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 Web 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.

More information:

Agent to Policy Server Communication (see page 60)

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 SiteMinder is protecting.

   number_of_policies_per_session

   Specifies the average number of SiteMinder policies that apply to a user during the session.

   Note: Each SiteMinder 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.

Note: For more information about configuring the Policy Server to log events to an audit database, see the *Policy Server Administration Guide.* For more information about configuring an audit database, see the *Policy Server Installation Guide*.

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

- Performance associated with authentication and authorization is affected because SiteMinder 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 Web Agent parameters and following the SiteMinder 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 Web Agents and Policy Servers, dynamic load balancing reduces latency and improves throughput because the Web Agents distribute requests among all of the Policy Servers.

More information:

Redundancy and High Availability (see page 60)

Data Tier Performance

Poor performance associated with SiteMinder data stores, especially user directories, is one of the most common reasons for poor SiteMinder 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 SiteMinder performance.
- The capacity under which your user directories have to operate. SiteMinder 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 SiteMinder 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 SiteMinder 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.

More information:

Capacity Planning Introduced (see page 75)

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 SiteMinder performance.

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

There are several general considerations to improving SiteMinder 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 SiteMinder 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 SiteMinder performance. If the user directory is operating at a high level of utilization, then no amount of SiteMinder 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 SiteMinder 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 SiteMinder can efficiently search users directories directly correlates with performance. Consider the following:

- Use directory indexing to enhance search results for SiteMinder:
  - 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 SiteMinder 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 SiteMinder performance is determining whether your user directories can handle this workload during sustained and peak periods of operation.

The following general factors influence SiteMinder 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.

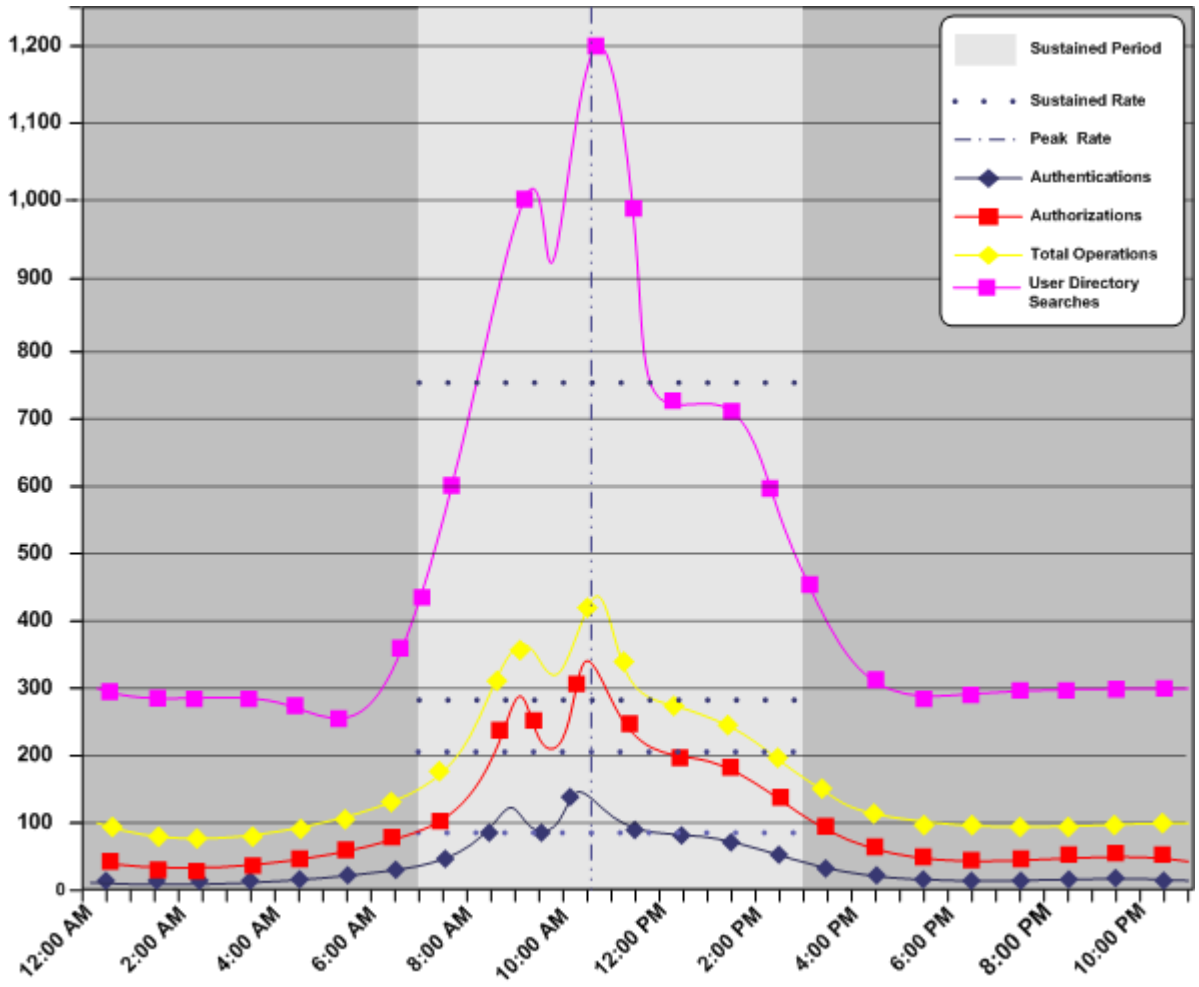
  In turn, the rate at which the Policy Server makes user directory requests to process the operations fluctuates. Some periods generate relatively few user directory requests, while others generate more.

  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.

The following graphic illustrates:

- The relationship between total operations and the user directory search rate.
- How each rate fluctuates throughout the day, is sustained for a specific period, and peaks within that period.



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.

More information:

Policy Server (see page 14)
How to Estimate a Sustained Authentication Rate (see page 77)
How to Estimate a Sustained Authorization Rate (see page 83)

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 SiteMinder policy determine:
  - If the SiteMinder policy membership filter results in one or more user directory searches.
  - If the responses bound to the SiteMinder policy results in one or more user directory searches.

More information:

Capacity Planning Introduced (see page 75)
SiteMinder Policy Membership and Authorization Performance (see page 142)
Responses and Authorization Performance (see page 142)

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 SiteMinder policy that is bound to a rule that fires when a user is authenticated (OnAuth rule).

  Note: For more information about configuring rules, see the *Policy Server Configuration Guide*. For more information about the relationship a rule has to a SiteMinder policy, see the *Policy Server Configuration Guide*.

- One search/query for each SiteMinder policy that is bound to a response that returns user attributes.

  Note: For more information about responses and their relationship to rules, see the *Policy Server Configuration Guide*.

- One write/update per user store enabled for Password Services. If Password Services does not apply to the user directories in the SiteMinder policy domain, a write/update is not required.

  Note: For more information about Password Services, see the *Policy Server Configuration Guide*.

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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder 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 SiteMinder policy membership, while others are dependent on SiteMinder policy design. You can estimate the number of Policy Server requests that each authorization creates using the following guidelines.

- One search/query for each SiteMinder 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 SiteMinder 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.

More information:

SiteMinder Policy Membership and Authorization Performance (see page 142)
Responses and Authorization Performance (see page 142)
User Authorization Cache (see page 143)

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 SiteMinder policy membership.
- Policies B may require up to two user directory requests to determine SiteMinder policy membership.
- Policies C may require up to three user directory requests to determine SiteMinder 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 SiteMinder 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 SiteMinder 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.

More information:

User Authorization Cache (see page 143)

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 SiteMinder 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 SiteMinder 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:

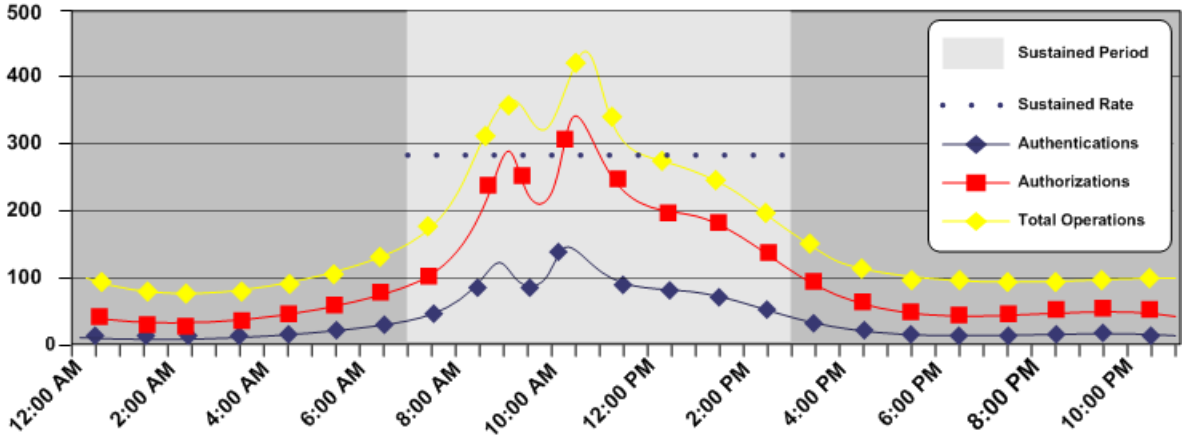
- A sustained period during which the system is servicing an average number of operations.
- How these requests translate into user directory searches.

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.

The following figure is an example of these metrics.



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.

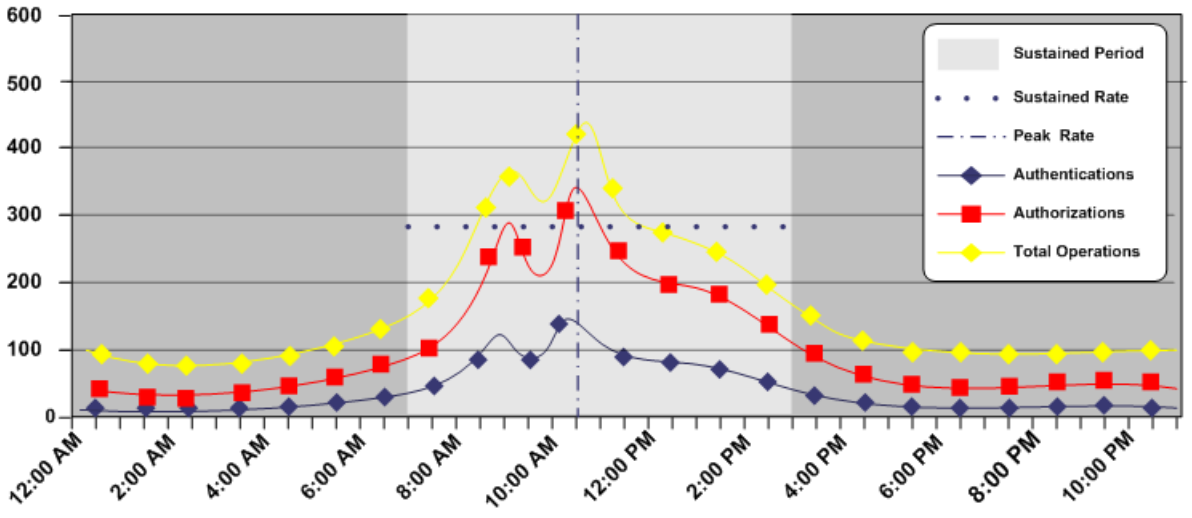
Estimate the Peak User Directory Search Rate

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.

The following figure is an example of these metrics:



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.

Chapter 6: Diagnose Implementation Issues

This section contains the following topics:

Diagnose Issues Introduced

The problems you can encounter during a SiteMinder implementation vary and are unique to each environment. Problems can be related to the deployment of individual components to the overall performance of the environment.

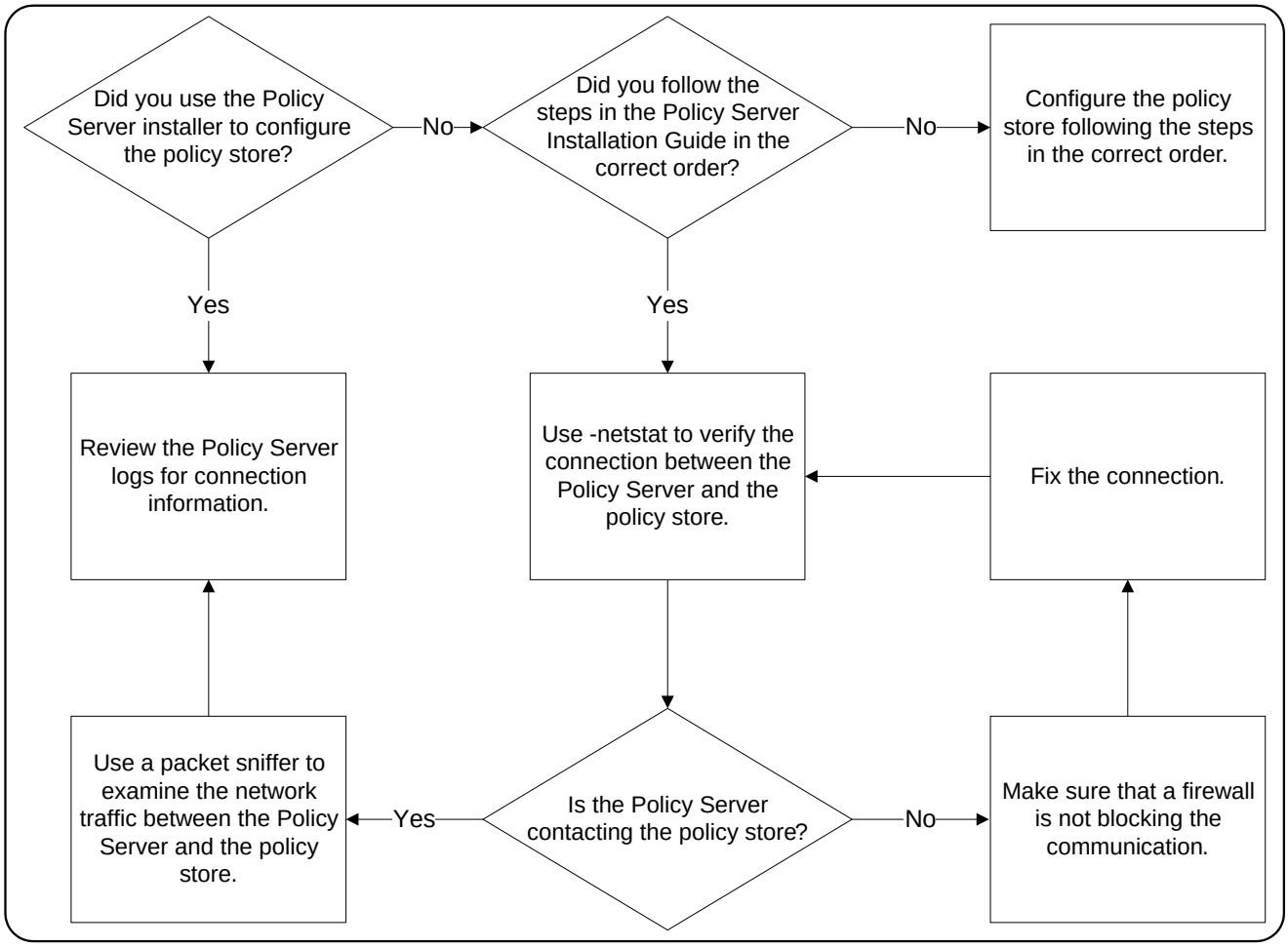
The following sections detail:

- How to diagnose common implementation issues
- How to work with Support to resolve issues efficiently
- Where to locate additional SiteMinder documentation to help troubleshoot issues
- Some tools that you can use to measure SiteMinder performance

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:



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

If you require assistance from the SiteMinder 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.

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 Web 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 Web 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 SiteMinder data stores.

- If applicable, any other CA or third–party products integrated with SiteMinder.
- 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 SiteMinder component or new hardware.
- When the problem started.

Note: You can use the SiteMinder Platform Support Matrix to verify that the issue is not related to an operating system or third–party product that SiteMinder does not support. For more information, see the SiteMinder Platform Support Matrix.

More information:

Locate the SiteMinder Platform Support Matrix (see page 179)

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

Consider the following:

- All Policy Server logs are located in *siteminder_home*\log.

  siteminder_home

  Specifies the Policy Server installation path.

  Note: For more information about configuring the Policy Server profiler, see the *Policy Server Administration Guide*. For more information about auditing, see the *Policy Server Administration Guide*.

- Web Agent logs have no default location or default names.

  Note: For more information about configuring Web Agent logging, see the *Web Agent Configuration Guide*.

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.

   Note: If you are attaching log files, be sure that the set of files match and that all are from the same time as when the issue occurred.

More information:

Environment Information (see page 167)
Log Files (see page 168)

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 SiteMinder data information file (smdif) to the ticket.

Note: For more information about exporting the policy store, see the *Policy Server Administration Guide*.

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.

SiteMinder 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.

Note: For more information about configuring the Policy Server profiler, see the *Policy Server Administration Guide*.

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

Web Agent Crash

If the Web Agent 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. 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:
   - (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 Web Agent log and the web server error log.

   Note: If you are attaching log files, be sure that the set of files match and that all are from the same time as when the issue occurred.
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. Attach the Web Agent trace log and the web server access log.

More information:

Environment Information (see page 167)
Log Files (see page 168)

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.
   - (Web Agent) If you are experiencing a Web Agent issue, attach the Web Agent log and the web server error log.

   Note: If you are attaching log files, be sure that the set of files match and that all are from the same time as when the issue occurred.

More information:

Environment Information (see page 167)
Log Files (see page 168)

Functional Issues

A functional issue is defined as an issue where SiteMinder 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:
   - (Policy Server) If you are experiencing a Policy Server issue, attach the Policy Server log and the Policy Server audit log.
   - (Web Agent) If you are experiencing a Web Agent issue, attach the Web Agent log and the web server error log.

   Note: If you are attaching log files, be sure that the set of files match and that all are from the same time as when the issue occurred.
4. Export the policy store to a SiteMinder data information file (smdif) and attach the file.

   Note: For more information about exporting the policy store, see the *Policy Server Administration Guide*.
5. Attach logs:
   - (Policy Server) If you are experiencing a Policy Server issue, attach the Policy Server profiler log.
   - (Web Agent) If you are experiencing a Web Agent issue, attach the Web Agent trace log and the web server access log.

More information:

Environment Information (see page 167)
Log Files (see page 168)

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:
   - (Policy Server) 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.
   - (Web Agent) If you are experiencing a Web Agent issue:
     - attach the Web Agent log with the point of failure, the web server error log with the point of failure, and the Web Agent trace log with the point of failure.
     - attach the Web Agent trace log with the system functioning correctly.

   Note: If you are attaching log files, be sure that the set of files match and that all are from the same time as when the issue occurred.

More information:

Environment Information (see page 167)
Log Files (see page 168)

Locate Knowledge Base Articles

The SiteMinder bookshelf is only one resource that is available to you. SiteMinder knowledge base (KB) articles are available on the CA Technical Support site. These articles address various topics related to managing and troubleshooting a SiteMinder environment.

To locate SiteMinder KB articles

1. Log into the Technical Support site.
2. Click Support by Product.

   The Support by Product page appears.
3. Locate CA SiteMinder in the product list and click the link.

   The CA SiteMinder 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 SiteMinder Performance

Measuring SiteMinder 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 SiteMinder 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 SiteMinder performance include:

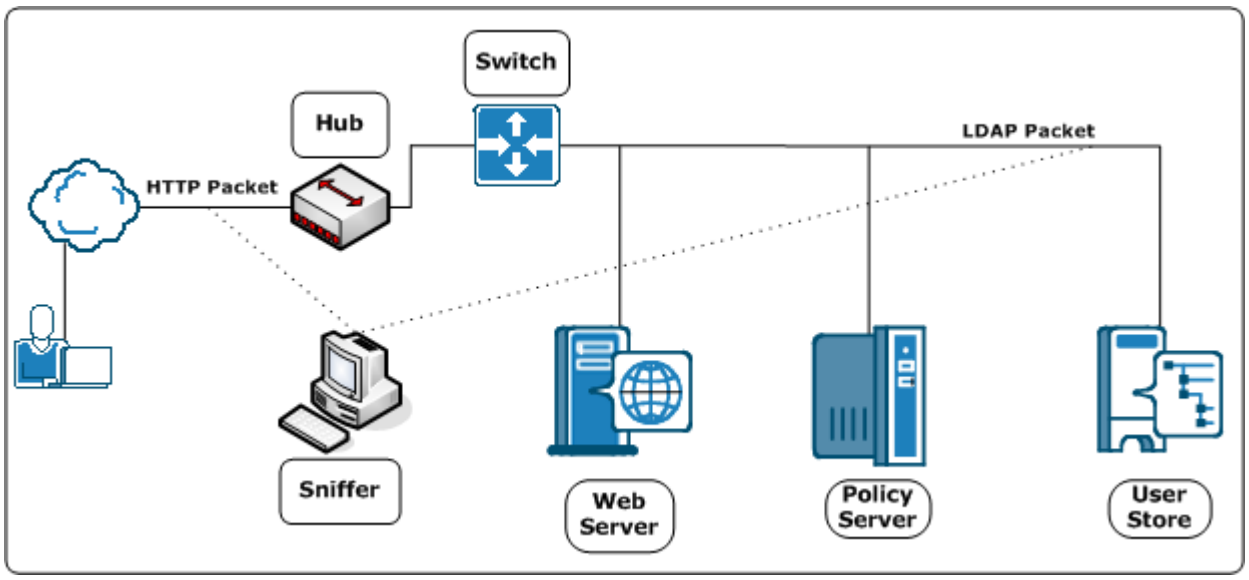
- Network sniffers
- The SiteMinder OneView Monitor
- The SiteMinder Test Tool
- Directory server utilities and SQL analyzers

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 SiteMinder deployment.



SiteMinder OneView Monitor

You can use the SiteMinder OneView Monitor to identify performance bottlenecks and gather metrics about resource usage in a SiteMinder deployment. The OneView Monitor also displays alerts when certain events, such as component failure, occur by collecting operational data from the following SiteMinder 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.

Note: For a complete list of the data types the OneView Monitor can provide, see the *Policy Server Administration Guide*. For more information about installing the OneView Monitor, see the *Policy Server Installation Guide.*

SiteMinder Test Tool

You can use the SiteMinder Test Tool utility to test the interaction between SiteMinder 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.

Note: For more information about the Test Tool utility, see the *Test Tool Help*.

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.

Appendix A: Platform Support and Installation Media

This section contains the following topics:

Locate the SiteMinder Platform Support Matrix

You can find a comprehensive list of the CA and third–party components supported by SiteMinder on the Technical Support site.

To locate the support matrix from the Support site

1. From the Technical Support site, click Enterprise/Small and Medium Business.

   The Support for Business and Partners screen appears.

2. Log in to CA Support Online.

   The CA Support Online Basic and Enterprise User screen appears.

3. Enter your login credentials, again.

   The CA Support Online screen appears.

4. Under Support, click Support By Product.

5. Select CA SiteMinder from the Select a Product Page list.

   The CA SiteMinder screen appears.

6. Scroll to the Product Status section and click CA SiteMinder Platform Support Matrices.

Note: You can download the latest JDK and JRE versions at the Sun Developer Network.

Locate the Bookshelf

The SiteMinder bookshelf is available on the

To locate the support matrix from the Support site

1. Go to the CA Technical Support site.
2. If the Get Support tab is not pulled to the front, click Get Support.
3. Under Find Product News and Support, click Product Pages.

   The Support by Product page appears.
4. Locate CA SiteMinder in the product list and click the link.

   The CA SiteMinder product page appears.
5. Click Bookshelves.
6. Click the link for the release that you require.

   The SiteMinder bookshelf main page appears.

Locate the Installation Media

You can find a comprehensive list of the SiteMinder installation media on the Technical Support site.

To locate the support matrix from the Support site

1. From the Technical Support site, click Enterprise/Small and Medium Business.

   The Support for Business and Partners screen appears.
2. Log in to CA Support Online.

   The CA Support Online Basic and Enterprise User screen appears.
3. Enter your login credentials, again.

   The CA Support Online screen appears.
4. Under Support, click Download Center, Products.

   The Download Center screen appears.
5. Type CA SiteMinder in the Select a Product field.
6. Select a release from the Select a Release list.

7. Select a service pack from the Select a Gen Level list.
8. Click Go.

   The Product Downloads screen appears. All SiteMinder installation executables are listed.

Index

F

G

H

I

K

L

M

N

O

P

R

S

T

U

W

Y