eTrust® SiteMinder™ Agent r6.0 for IBM WebSphere®

SiteMinder Agent for IBM WebSphere Guide

r6.0

CA Product References

This document references the following CA products:

- CA eTrust® SiteMinder™

Contact Technical Support

For online technical assistance and a complete list of locations, primary service hours, and telephone numbers, contact Technical Support at http://ca.com/support.

Contents

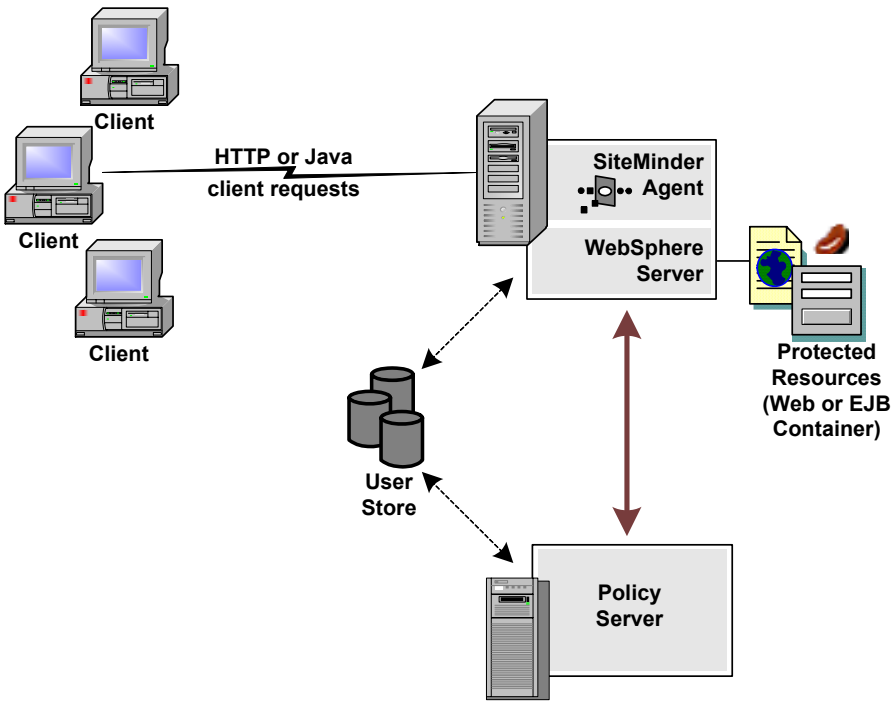
Chapter 1: Introduction

The SiteMinder Agent for IBM WebSphere provides a complete SiteMinder-based access control solution for IBM WebSphere Application Server 6.0. The SiteMinder Agent integrates the WebSphere Application Server into the SiteMinder environment, enabling you to implement policy-based access control to protect your WebSphere-hosted Web applications and Enterprise JavaBeans (EJB) resources.

This section contains the following topics:

Overview

The SiteMinder Agent for IBM WebSphere resides in a WebSphere Application Server, enabling you to extend the SiteMinder environment to protect WebSphere-hosted resources (in the Web and EJB containers), as shown in the following high-level example environment.



The SiteMinder Agent for IBM WebSphere provides the following features:

- SiteMinder Integration with the J2EE platform
- Fine-grained access control of the following J2EE resources:
  - Web Applications (including servlets, HTML pages, JSP, image files)
  - EJB components
- Support for bi-directional SiteMinder and WebSphere single sign-on (SSO)
- Support for WebSphere clustering

The SiteMinder Agent additionally supports:

- J2EE RunAs identity
- EJB stand-alone client applications
- Multi-byte character usernames
- User mapping to support environments in which WebSphere and SiteMinder are not configured to use the same user store
- Centralized and dynamic agent configurations
- Caching of resource protection decisions and authentication and authorization decisions
- Web application error page processing (so that failure to answer an authentication request results in redirection to an error page)
- Logging
- Authorization auditing

Required Background Information

This guide assumes that you have the following technical knowledge:

- An understanding of Java, J2EE standards, J2EE application servers, and multi-tier architecture
- A strong knowledge of Java technology, including:
  - Servlets
  - Java Server Pages (JSP)
  - Enterprise JavaBeans (EJB)
  - J2EE Web Applications
- Experience with the IBM WebSphere Application Server Version 6.0.x, its architecture and security infrastructure.
- Familiarity with Java Authentication and Authorization Server (JAAS) and other WebSphere security-related topics:
  - WebSphere Trust Association Interceptor (TAI) concepts
  - Login modules
  - Java Authorization Contract for Containers (JACC) specification (JSR-115)

- Familiarity with SiteMinder concepts, terms, and Policy Server configuration tasks
- Familiarity with SiteMinder Web Agents

Additionally, to effectively plan your security infrastructure, you must be familiar with the applications that you plan to protect with SiteMinder.

SiteMinder Agent for IBM WebSphere Components

The SiteMinder Agent for IBM WebSphere consists of three custom Agent modules that plug into WebSphere's security infrastructure.

SiteMinder Trust Association Interceptor (TAI)

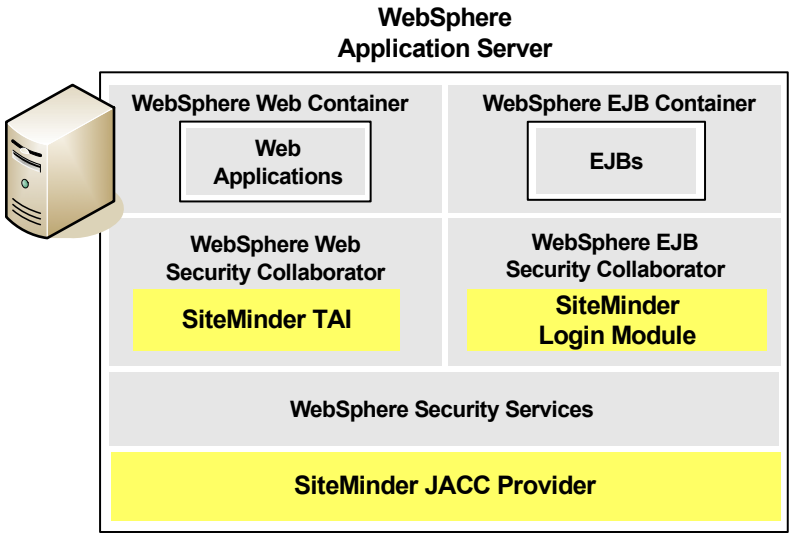
Establishes a Web Trust Association between WebSphere and SiteMinder so that credentials obtained from HTTP requests for Web container resources can be validated against associated user directories configured in SiteMinder. Populates the Subject with a *SiteMinder Principal* that can be used by the SiteMinder JACC Provider for authorization.

SiteMinder Login Module

Validates user credentials obtained from Java client requests and system logins against associated user directories configured in SiteMinder. Populates the Subject with a SiteMinder Principal that can be used by the SiteMinder JACC Provider for authorization.

SiteMinder Java Authorization Contract for Containers (JACC) Provider

Provides SiteMinder policy-based authorization decisions for requests for Web or EJB resources using credentials in an associated SiteMinder Principal placed in the subject by the SiteMinder TAI or SiteMinder Login Module.

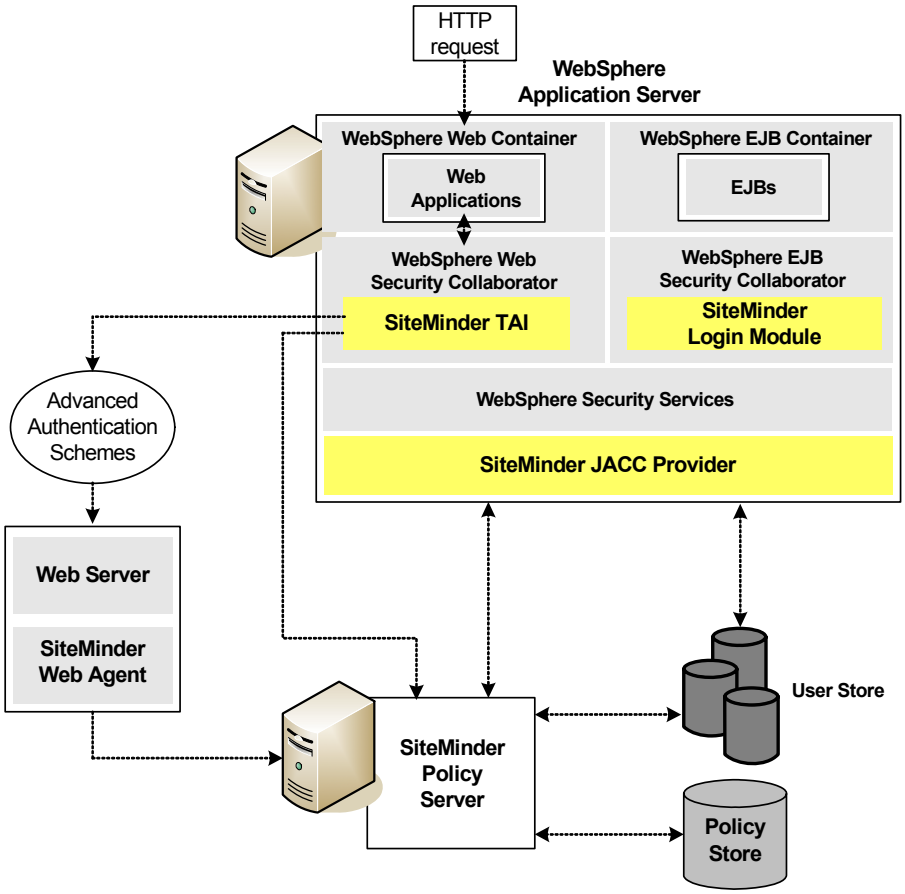


SiteMinder Trust Association Interceptor (TAI)

The SiteMinder Trust Association Interceptor module is a SiteMinder security module that plugs into the WebSphere TAI public security interface to provide a Web Trust Association (WTA) between WebSphere and SiteMinder. In this WTA, WebSphere assigns the SiteMinder TAI the responsibility of validating HTTP requests for Web container resources and creating principals that establish identity and can be used for authorization by the SiteMinder JACC Provider.

The SiteMinder TAI handles requests for HTTP resources:

- From users with pre-established SiteMinder sessions without challenging them for credentials (validating the session and obtaining user names from the associated SiteMinder session ticket cookies).
- From users without pre-established SiteMinder sessions by challenging them for credentials using SiteMinder basic or advanced authentication schemes. A SiteMinder Web Agent provides authentication services for advanced authentication schemes.



The SiteMinder TAI always validates requests which contain SiteMinder session cookies; you must configure it to challenge other requests for credentials.

If SiteMinder authentication is successful, the SiteMinder TAI populates a JAAS Subject with a SiteMinder Principal that contains the username of the authenticated user and associated SiteMinder session data. Additionally, the SiteMinder TAI propagates the identity of the authenticated user to WebSphere, which then creates its own principal and adds it to the Subject for use by other, non-SiteMinder security modules.

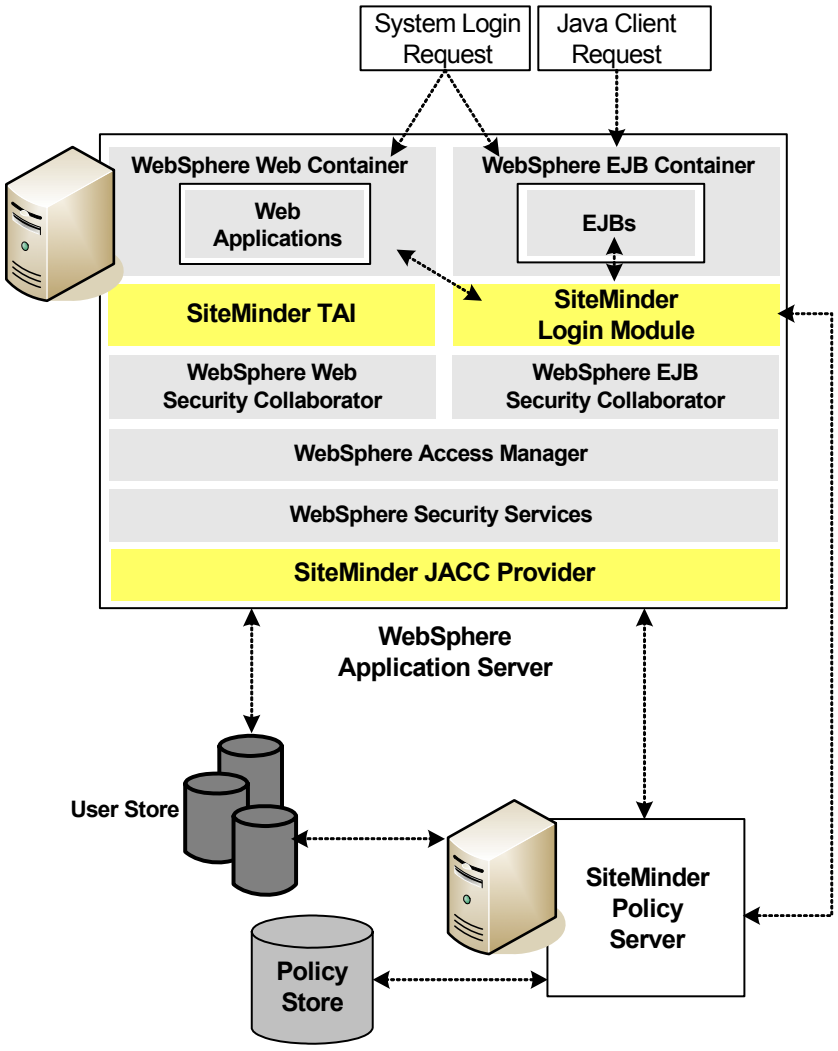
Note: If the SiteMinder TAI is configured to support environments in which the Policy Server and WebSphere have separate user stores, the SiteMinder TAI propagates to WebSphere a mapped user identity that matches an entry in the WebSphere user store.

More information:

Choosing the Agent Configuration You Need (see page 21)
Configuring the TAI, SiteMinder-Side (see page 57)
Configuring the SiteMinder TAI in WebSphere (see page 75)
Identity and User Mapping (see page 18)

SiteMinder Login Module

The SiteMinder Login Module is a standard JAAS Login Module that authenticates credentials (username/password) obtained from Java client and system login requests.



If SiteMinder authentication is successful, the SiteMinder Login Module populates a JAAS Subject with a SiteMinder Principal that contains the username and associated SiteMinder session data. Additionally, the SiteMinder Login Module propagates the identity of the authenticated user to WebSphere, which then creates its own principal and adds it to the Subject.

Note: If the SiteMinder Login Module is configured to support environments in which the Policy Server and WebSphere have separate user stores, the SiteMinder Login Module propagates a mapped user identity that matches an entry in the WebSphere user store to the WebSphere Application Server.

More information:

Choosing the Agent Configuration You Need (see page 21)
Configuring the Login Module, SiteMinder-Side (see page 64)
Configuring the Login Module in WebSphere (see page 77)
Identity and User Mapping (see page 18)

Request Types Supported by the SiteMinder Login Module

The SiteMinder Login Module handles the following request types:

- Java client (RMI-IIOP) requests for EJB container resources
- System login (such as J2EE RunAs identity) requests for resources in Web and EJB containers

More information:

J2EE Programmatic Security Call Principal Usage (see page 19)

Recreating Subjects by Asserting WebSphere Propagation Tokens

Note: If you are running IBM WebSphere Application Server v6.0.2.7 (or later) you can ignore this section; the issues it describes are not applicable in your deployment.

In certain situations, WebSphere must recreate Subjects (including those containing SiteMinder Principals) from a WebSphere propagation token, for example when:

- Requests are moved between servers in WebSphere cluster configurations
- Requests are moved between servers in WebSphere SSO configurations
- A WebSphere Application Server shuts down during an active user session

HTTP requests are handled and populated with a SiteMinder Principal by the SiteMinder TAI on the server on which the request was initially received. However, versions of WebSphere prior to 6.0.2.7 do not invoke TAI modules in subject recreation situations, relying rather on configured Login Modules.

This issue is resolved in WebSphere v6.0.2.7. If you have an earlier version, the SiteMinder Login Module is required to recreate Subjects initially populated with a SiteMinder Principal in all SiteMinder Agent for IBM WebSphere deployments that include the SiteMinder JACC Provider—regardless of whether you need the Login Module to handle authentication requests for EJB container resources. Configuration of the SiteMinder Login Module for this purpose is also recommended in what would otherwise be a TAI-only environment in a WebSphere SSO configuration.

Note: For more information about WebSphere propagation tokens, search for "Security Attribute Propagation" in the IBM WebSphere Application Server online documentation.

More information:

Choosing the Agent Configuration You Need (see page 21)
Adding the SiteMinder Login Module as a WebSphere WEB_INBOUND Login Module (see page 79)

SiteMinder Java Authorization Contract for Containers (JACC) Provider

The SiteMinder JACC Provider is a JAAS module that implements the Java Authorization Contract for Containers (JSR-115) specification, enabling the SiteMinder Agent for IBM WebSphere to handle authorization decisions for WebSphere Web and EJB resources.

The SiteMinder JACC Provider determines whether an authenticated user is allowed to access a protected WebSphere resource, based on associated SiteMinder policies configured using the Policy Server User Interface.

The SiteMinder JACC Provider only accepts Subjects populated with a SiteMinder Principal containing SiteMinder session data (required to prove that SiteMinder authentication has occurred).

The SiteMinder JACC Provider implements the interfaces defined in the JSR-115 specification and fulfills the following contracts (with certain limitations):

- Provider Configuration Subcontract
- Policy Decision and Enforcement Subcontract

The SiteMinder JACC Provider does *not* comply with the JSR-115 Policy Configuration Subcontract; it does not create policies for applications. Security policies for applications must therefore be created by SiteMinder administrators using the Policy Server User Interface.

More information:

Choosing the Agent Configuration You Need (see page 21)
Configuring the SiteMinder JACC Provider, SiteMinder-Side (see page 69)
Configuring the SiteMinder JACC Provider in WebSphere (see page 81)
Configuring Policies for the SiteMinder Agent (see page 91)

Other Deployment Considerations

Other factors to consider when planning your SiteMinder Agent for IBM WebSphere deployment are:

- **Identity and User Mapping (see page 18)—**Required if the environment needs user mapping to provide WebSphere with user identities that match those in its user store when SiteMinder and WebSphere are not configured with the same user directories.
- **J2EE Programmatic Security (see page 19)—**Configuration requirements and considerations associated with SiteMinder Agent for IBM WebSphere support for J2EE programmatic security API calls.
- **User Session Handling (see page 19)—**Steps you must take to resolve user session synchronization issues because SiteMinder and WebSphere handle user sessions differently.
- **SiteMinder API Changes (see page 20)—**Changes you must make for client applications that use the SiteMinder Agent API.

Identity and User Mapping

The SiteMinder Agent for IBM WebSphere provides user mapping functionality that enables the SiteMinder Agent for IBM WebSphere to support environments in which SiteMinder is responsible for user authentication, but SiteMinder and WebSphere are not configured to authenticate users against the same user store.

By default, both the SiteMinder TAI and SiteMinder Login Module are responsible for authenticating the user against SiteMinder and propagating the user's identity by populating the Subject with a SiteMinder Principal required to authorize the user using the SiteMinder JACC Provider. Additionally, they propagate that user identity to WebSphere, which creates its own principal and places that principal in the Subject.

However, WebSphere *requires* that an identity that is valid against WebSphere's user registry is available in the Subject to handle WebSphere Single Signon (SSO) and all J2EE programmatic security calls. Exceptions to this are isUserInRole() and isCallerInRole(), which are handled by the JACC specification and thus require only the SiteMinder Principal.

To handle this requirement, you configure user mapping policy objects (a user mapping rule, response, and policy) in the policy realm of the SiteMinder TAI and SiteMinder Login Module. These objects define a mapped identity that is valid against the WebSphere user registry. Then, when users make requests, they are authenticated using the SiteMinder identity, but the SiteMinder Agent for IBM WebSphere module responsible for authentication propagates an alternate, mapped user identity that WebSphere converts into a principal and places in the Subject in addition to the SiteMinder Principal.

More information:

Configuring SiteMinder Policies to Support User Mapping (Optional) (see page 97)

User Session Handling

SiteMinder and WebSphere handle user sessions differently. To synchronize sessions, you must perform some additional configuration steps.

More information:

Session Handling (see page 105)

J2EE Programmatic Security Call Principal Usage

J2EE application components have access to standard security APIs that provide user identity and role membership information used for program logic. There are two types of calls—one that returns the identity of the user and another that returns Boolean decisions, based on an input role indicating whether the user has membership in that role.

API Call	Handling Container	Description
getRemoteUser ()	Web	Returns the login identity of the user making a request if the user has been authenticated, or null if the user has not been authenticated.

API Call	Handling Container	Description
getUserPrincipal ()	Web	Returns a java.security.Principal object containing the name of the current authenticated user.
isUserInRole (String role)	Web	Returns a Boolean indicating whether the authenticated user is included in the specified logical role.
getCallerPrincipal ()	EJB	Returns a java.security.Principal object containing the name of the caller.
isCallerInRole (String role)	EJB	Returns a Boolean indicating whether the caller is included in the specified logical role.

WebSphere always uses its own identity Principal to answer J2EE programmatic security calls (except isUserInRole() and isCallerInRole(), which use the SiteMinder Principal.

Note: The SiteMinder Agent for IBM WebSphere supports only globally-scoped roles; it does not support roles scoped to an application for any J2EE programmatic calls.

SiteMinder Agent API Changes

This release internally replaces the JNI-based SiteMinder Agent API with a pure Java version that is not yet available for external use.

The public facing API classes have not changed and are deployed in smagentapi.jar in *WS_HOME*/lib/ext.

Therefore, any client applications that use the SiteMinder Agent API must ensure that the API jar file (smjavaagentapi.jar) is placed ahead of the pure Java version (smagent.jar) in the application's classpath. It must be placed ahead only in the classpath of the application itself, not for deployed SiteMinder Agent modules.

Choosing the Agent Configuration You Need

Although all the SiteMinder Agent for IBM WebSphere modules are installed by the Agent installation, you do not need to configure all of them. The following table provides an overview of the SiteMinder Agent modules, their functions and interdependencies.

Agent Component/Function	Upstream Requirements	Downstream Requirements
SiteMinder TAI (no challenge for credentials) (Web container authentication; SiteMinder pre-authenticated requests only)	A trusted issuer of SiteMinder session cookies	None for authentication-only solution. To support SiteMinder authorization, SiteMinder JACC Provider required; SiteMinder Login Module may be required to assert WebSphere propagation tokens (see page 16) in Subject recreation situations.
SiteMinder TAI (challenge for credentials) (Web container authentication; all requests)	SiteMinder Web Agent for non-basic authentication schemes	None for authentication-only solution. To support SiteMinder authorization, SiteMinder JACC Provider required; SiteMinder Login Module may be required to assert WebSphere propagation tokens (see page 16) in Subject recreation situations.
SiteMinder Login Module (EJB container and system login authentication; assertion of WebSphere propagation tokens)	None	To support SiteMinder authorization, SiteMinder JACC Provider required; otherwise user mapping must be configured to provide WebSphere principal for use by WebSphere security.

Agent Component/Function	Upstream Requirements	Downstream Requirements
SiteMinder JACC Provider (Authorization)	Subject populated with SiteMinder Principal. **Note:** To ensure the validity of the SiteMinder Principal (see page 16) in Subject recreation situations, the SiteMinder Login Module is required in all SiteMinder JACC Provider-equipped configurations of WebSphere releases before v.6.0.2.7.	None

While the previous table shows that a range of different Agent module configurations is possible, two configurations are most likely to provide the solutions to real-life deployment scenarios:

Requirement	Suggested Configuration
You need to establish a trust relationship between the SiteMinder and WebSphere Single Signon (SSO) environments so that HTTP clients authenticated by SiteMinder are not re-challenged by WebSphere when they access Web applications hosted by a WebSphere Application Server or the converse. (Or you are upgrading from an existing SiteMinder Application Server Agent for WebSphere solution.) You have existing WebSphere or application-based authorization policies that are sufficient for your needs.	Configure the SiteMinder TAI in a Web Trust Association environment in which: ▪ HTTP requests to Web applications are intercepted by the SiteMinder TAI ▪ Users are authenticated through policies defined on the Policy Server In a WebSphere SSO environment, you may require the SiteMinder Login Module to assert WebSphere propagation tokens (see page 16) in situations when WebSphere must reestablish Subjects created by the SiteMinder TAI.

Requirement	Suggested Configuration
You need to establish a trust relationship between the SiteMinder and WebSphere Single Signon (SSO) environments so that HTTP clients authenticated by SiteMinder are not re-challenged by WebSphere when they access Web applications hosted by a WebSphere Application Server or vice versa. You want to implement SiteMinder authentication and authorization policies for requests for Web and/or EJB client applications.	Configure the complete SiteMinder Agent solution, comprising: ■ SiteMinder TAI ■ SiteMinder Login Module ■ SiteMinder JACC Provider

Use Cases

The SiteMinder Agent for IBM WebSphere modules that you configure depend upon your requirements and fall into the two scenarios described in Choosing the Agent Configuration You Need (see page 21):

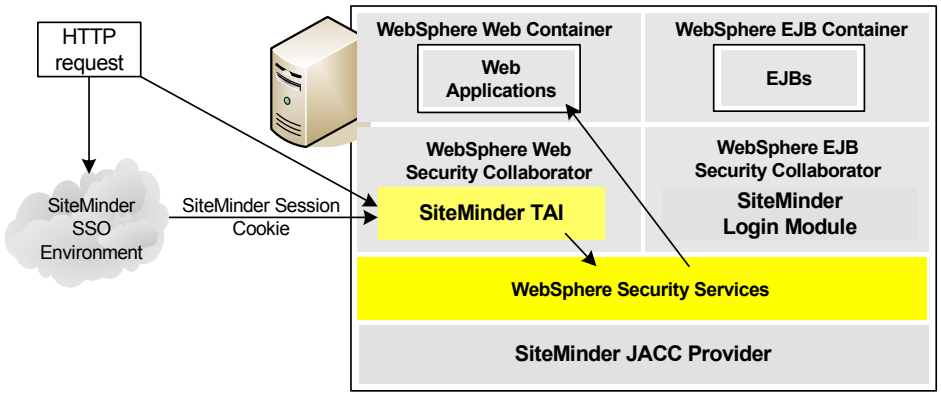
- SiteMinder TAI-Only Use Case
- All SiteMinder Agent for IBM WebSphere Modules Use Case

SiteMinder TAI-Only Use Case

The SiteMinder-TAI only use case lets you combine SiteMinder and WebSphere single sign-on environments. In this scenario, users authenticated within the SiteMinder environment are allowed access to WebSphere-hosted Web applications without being challenged by WebSphere.

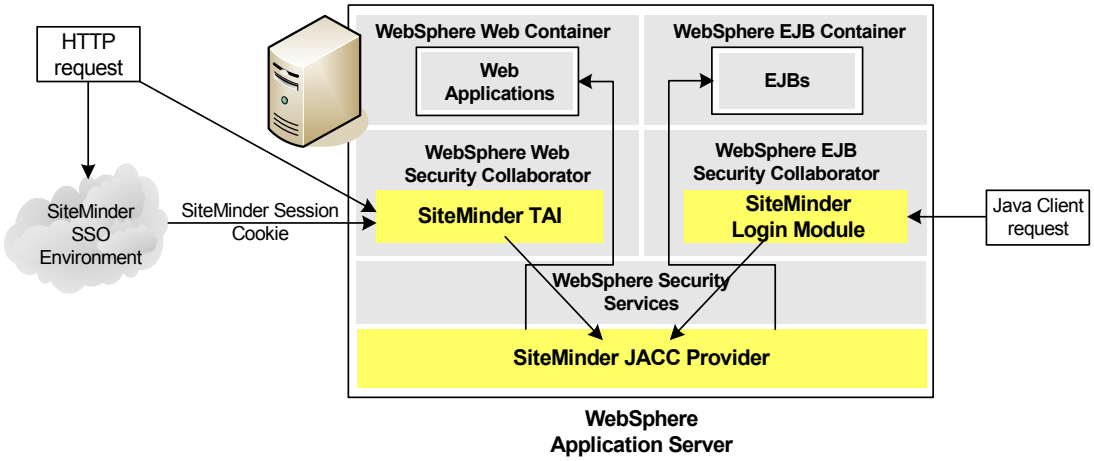
You can also configure the SiteMinder TAI to handle requests without associated SiteMinder session cookies by challenging them for credentials and authenticating them against SiteMinder user directories.

Authorization is performed using existing WebSphere security policies.



All Modules Use Case

The use case illustrated in the following diagram enables you to handle all the request types supported by the SiteMinder TAI and the SiteMinder Login Module and provides SiteMinder authorization using the SiteMinder JACC Provider.



The SiteMinder TAI handles requests for Web container applications (with or without associated SiteMinder session cookies if configured to challenge for credentials).

The SiteMinder Login Module handles Java client requests for EJB container resources and J2SE RunAs requests for resources in either container.

The SiteMinder JACC Provider provides SiteMinder authorization for all requests.

Recommended Reading List

To learn about the WebSphere Application Server and Java, see the following resources:

- IBM Redbooks Online

  http://www.redbooks.ibm.com/Redbooks.nsf/redbooks/ (http://www.redbooks.ibm.com/redbooks.nsf/redbooks/)

- IBM WebSphere Application Server Information Center

  http://www-306.ibm.com/software/webservers/appserv/was/ (http://www-306.ibm.com/software/webservers/appserv/was/)

- Sun Microsystems, Inc., online documentation

  http://java.sun.com (http://java.sun.com).

Chapter 2: Preconfiguring Policy Objects for the SiteMinder Agent

This section contains the following topics:

Policy Object Preconfiguration Overview

Before you install the SiteMinder Agent for IBM WebSphere, the SiteMinder Policy Server must be installed and be able to communicate with the system where you plan to install the SiteMinder Agent. Additionally, you must configure the Policy Server with the following:

- **A SiteMinder administrator that has the right to register trusted hosts**

  A trusted host is a client computer where one or more SiteMinder Agents are installed. The term trusted host refers to the physical system. There must be an administrator with the privilege to register trusted hosts.

  To configure an administrator, see the Administrators chapter of *CA eTrust Policy Design*.

- **Agent object/Agent identity**

  An Agent object creates an Agent identity by assigning the Agent a name. You define an Agent identity from the Agents object in the Policy Server User interface. You assign the Agent identity a name and specify the Agent type as a Web Agent.

  The name you assign for the Agent is the same name you specify in the DefaultAgentName parameter for the Agent Configuration Object that you must also define to centrally manage an Agent.

- **Host Configuration Object**

  This object defines the communication between the trusted host and the Policy Server after the initial connection between the two is made.

  A trusted host is a client computer where one or more SiteMinder Agents can be installed. The term trusted host refers to the physical system, in this case the WebSphere Application Server host.

  Do not confuse this object with the trusted host's configuration file, SmHost.conf, which is installed at the trusted host after a successful host registration. The settings in the SmHost.conf file enable the host to connect to a Policy Server for the first connection only. Subsequent connections are governed by the Host Configuration Object.

  For more information, see *CA eTrust Policy Design*.

- **Agent Configuration Object**

  This object includes the parameters that define the SiteMinder Agent configuration. There are a few required parameters you must set for basic operation.

  The Agent Configuration Object must include a value for the DefaultAgentName parameter. This entry should match an entry you defined in the Agent object.

  For more information, see *CA eTrust Policy Design.*

Note: If you are using the SiteMinder Agent for IBM WebSphere to challenge for credentials using an advanced authentication scheme, you must also configure the policy objects for the Web Agent that performs authentication.

For detailed information about how to configure SiteMinder Agent-related objects, see *CA eTrust SiteMinder Policy Design, eTrust SiteMinder Web Agent Guide,* and the *eTrust SiteMinder Web Agent Installation Guide*.

Preconfiguring the Policy Objects

The following is an overview of the configuration procedures you must perform on the Policy Server prior to installing the Agent software:

1. Duplicate or create a new Host Configuration Object, which holds initialization parameters for a Trusted Host. (If upgrading from an earlier Agent install, you can use the existing Host Configuration object).

   The Trusted Host is a server that hosts one or more Agents and handles their connection to the Policy Server.

2. As necessary, add or edit Trusted Host parameters in the Host Configuration Object that you just created.

3. Create an Agent identity for the SiteMinder Agent for WebSphere. You must select **Web Agent** as the Agent type for the SiteMinder Agent for IBM WebSphere and its constituent modules.
4. Duplicate an existing or create a new Agent Configuration Object, which holds Agent configuration parameters and can be used to centrally configure a group of Agents.
5. In the Agent Configuration Object you just created, ensure that the DefaultAgentName parameter is set to specify the Agent identity defined in Step 3.

Note: You can optimize the Agent configuration after installation. For example, you can create additional Agent Configuration Objects to provide per-module configuration and logging options as described in Fine-Tuning Your Agent Configuration Environment (see page 52).

What to Do After Preconfiguring the Policy Server

After preconfiguring the Policy Server for the Agent, install the SiteMinder Agent for IBM WebSphere software as described in Installing and Upgrading the Agent.

Chapter 3: Installing and Upgrading the Agent

This chapter describes how to install the SiteMinder Agent for IBM WebSphere on Windows and UNIX platforms. The SiteMinder Agent installation includes the following modules:

- SiteMinder Trust Association Interceptor (TAI)
- SiteMinder Login Module
- SiteMinder Java Authorization Contract for Containers (JACC) Provider

Note: Although all Agent modules are installed when you run the Agent installation, you need only configure the modules that you require.

This section contains the following topics:

Upgrading from a Previous Release

The SiteMinder Agent for IBM WebSphere software cannot be upgraded from a previous version. To install the current version, you must first uninstall the previous version of SiteMinder Application Server Agent for IBM WebSphere. For information, see the Agent Guide associated with the release that you need to uninstall.

However, if you are upgrading from the previous SiteMinder TAI release, you can use most of your existing SiteMinder and WebSphere configuration settings that relate to the SiteMinder TAI. Any required changes are noted.

Before You Begin

This section describes the steps you must take before you install the SiteMinder Agent for IBM WebSphere.

Software Requirements

Before installing the SiteMinder Agent, install the following software:

Note: Be sure to install the prerequisite software in the correct order (see page 33).

- IBM WebSphere Application Server, Version 6.x and any cumulative fixes for this application server. For WebSphere hardware and software requirements, see the WebSphere documentation.
- SiteMinder Policy Server

To use the SiteMinder TAI to challenge Web requests that do not include a valid SiteMinder session cookie for credentials using advanced (other than Basic) authentication schemes:

- SiteMinder Web Agent

Note: The SiteMinder Policy Server and Web Agent (where applicable) can be installed on a different systems than the WebSphere Application Server.

For supported SiteMinder Policy Server and Agent versions and compatibility, go to the SiteMinder Support site (https://support.netegrity.com) and search for SiteMinder Platform Support Matrices.

More information:

Required Software Patches (see page 32)
Installation Checklist (see page 33)

Required Software Patches

Java Virtual Machine

The JVM required for use by the SiteMinder Agent Installation and IBM WebSphere must be patched to support unlimited key strength in the Java Cryptography Extension (JCE) package.

WebSphere's 1.4.x IBM JVMs are based on Sun's JVM for HP and Solaris platforms; these patches are available at Sun's website. The patches for all other SiteMinder supported platforms are available at IBM's website. See the IBM documentation for more details.

If the JVM is not patched to support unlimited key strength, host registration will fail during SiteMinder Agent installation and WebSphere will fail to start once the SiteMinder Agent has been configured on WebSphere.

Installation Checklist

Before you install the SiteMinder Agent for IBM WebSphere on the WebSphere server, complete the steps in the following table. To ensure proper configuration, follow the steps in order. You can place a check in the first column as you complete each step.

Completed?	Steps	For information, see...
	Install and configure the SiteMinder Policy Server.	*CA eTrust SiteMinder Policy Server Installation Guide*
	Install the IBM WebSphere Application Server.	The IBM WebSphere Application Server Documentation
	Patch JVMs for unlimited cryptography with the Java Cryptography Extension (JCE) package.	Required Software Patches (see page 32)
	Configure the Policy Server for the SiteMinder Agent for IBM WebSphere.	Preconfiguring Policy Objects for the SiteMinder Agent (see page 27)
	Install the Agent on the WebSphere Application Server. Note: For WebSphere clusters, install the Agent on each node in the cluster.	Installing the SiteMinder Agent for WebSphere (see page 34)
	Install and configure a SiteMinder Web Agent if using the SiteMinder TAI to challenge requests for credentials using advanced authentication schemes.	Installing a Web Agent to Process Advanced TAI Authentication (see page 41)

Setting a PATH Variable to the JVM on UNIX Systems

On UNIX systems, if your Java Virtual Machine (JVM) is not in the PATH variable, run these two commands:

```
PATH=$PATH:JRE
export PATH
```

JRE

Defines the location of your Java Runtime Environment bin directory. For example:

```
/opt/WebSphere/AppServer/java/jre/bin
```

Note: The SiteMinder Agent for IBM WebSphere requires that certain JVM 1.4 versions be patched to support unlimited key strength in their Java Cryptography Extension (JCE) packages (see page 32).

Installation Location References

In this guide:

- *ASA_HOME* refers to the installed location of the SiteMinder Agent for IBM WebSphere.
- *WS_HOME* refers to the installed location of the WebSphere Application Server.

Installing the SiteMinder Agent for IBM WebSphere

This section describes how to install the SiteMinder Agent for IBM WebSphere.

Information Required During Installation

The installation program prompts you for the following information:

- Location where WebSphere Application Server is installed. The default is:

  Windows: c:\Program Files\WebSphere\AppServer

  UNIX: /opt/Websphere/AppServer
- Policy Server IP Address
- If registering a new Trusted Host during installation (optional):
  - SiteMinder administrator user name and password
  - Unique Trusted Host Name.
  - Host Configuration Object name for the SiteMinder Agent

    (Object must already exist on the Policy Server before you install the SiteMinder Agent.)

  If you choose not to register the Trusted Host now, you can do it later (see page 41).
- If install system is already registered as a (SiteMinder 6.x) Trusted Host, the location of an existing Trusted Host configuration (SmHost.conf) file.
- SiteMinder Agent Configuration Object name.

  (This object must already exist on the Policy Server User before installing the SiteMinder Agent.)

Running the Installation in GUI Mode

To install the SiteMinder Agent for IBM WebSphere by using the graphical user interface (GUI) mode:

1. Start the SiteMinder Policy Server process, if it is not already running. (The installation program connects to the Policy Server to create a trusted host.)
2. Close all other programs.
3. As the user who installed WebSphere, connect to the system where WebSphere is installed. For example, if you installed as root, connect as root.
4. Download the following installation file to a temporary location:

   Windows: ca-asa-6.0-was-win32.exe

   UNIX (Solaris, HP-UX, Linux): ca-asa-6.0-was-unix.bin
5. On UNIX systems, depending on your permissions, you might need to add executable permissions to the installation file. For example:
   chmod +x ca-asa-6.0-was-unix.bin
6. Start the installer application by opening a command window, navigating to the temporary location, and entering:

   Windows: ca-asa-6.0-was-win32.exe
   (or you can double-click the file name in Windows Explorer)

   UNIX: sh ./ca-asa-6.0-was-unix.bin
7. Read the License Agreement. If you accept the terms, select the I accept the terms of the License Agreement option and click Next.
8. On the Choose Install Folder panel, specify a location for installing the SiteMinder Agent for IBM WebSphere and click Next. CA recommends the following default location:

   Windows: *drive*:\smwasasa

   UNIX: /opt/smwasasa

   If you specify a folder that does not exist, the installer asks if you want to create it. Click Yes to create it; the installer creates a folder named smwasasa in whatever directory you specify.

   The program installs the required files.

9. In the Choose WebSphere Folder dialog, specify the installation location of the WebSphere Application Server and click Install. For example:

   Windows: *drive*:\WebSphere\AppServer

   UNIX: /opt/WebSphere/AppServer

   The program installs the required files.

   Note: If the location you specify is not present, the installation program displays an error message and asks you to re-enter the information.

10. In the Host Registration dialog, select one of the following:

    - Yes, create trusted host — The installer invokes the Host Registration tool, smreghost, to register the unique trusted host name with the Policy Server and create the SmHost.conf file. Registering the system as a trusted host enables the SiteMinder Agent to establish a secure, trusted connection with the Policy Server.

      Before registering a trusted host, you must create a Host Configuration Object (see page 28) in the Policy Server.

    - No, use existing file — The installer invokes the smreghost tool to use an existing SmHost.conf file to establish the connection between the trusted host and the Policy Server.

      Note: Specify this option *only* if you are the reinstalling the SiteMinder Agent for WebSphere and the SmHost.conf file that you want to use was therefore created by the smreghost tool supplied with this release. The SiteMinder Agent for WebSphere is implemented using a pure Java SiteMinder Agent API and cannot use an SmHost.conf file created for another SiteMinder Agent to establish its connection to the Policy Server.

11. Do one of the following, then click Next:

If you selected...	Then...
Yes, create a trusted host	In the Host Registration dialog, enter the following information: ▪ Policy Server IP Address—IP address of the Policy Server where you are registering the host ▪ SM Admin Username—Name of the administrator permitted to register the host with the Policy Server ▪ SM Admin Username—Password for the SM Admin account ▪ Host Name—Unique name that represents the trusted host to the Policy Server. The name does not have to be the same as the physical client system you are registering; it can be any unique name. ▪ Host Config Object— Name of the Host Configuration Object specified in the Policy Server. See Creating a Host Configuration Object (see page 28). The installation program registers your unique trusted host name with the Policy Server. If your Policy Server is not running, an error message appears and you can register the trusted host later (see page 41). If you have not patched the JVM Java Cryptography Extension (JCE) package for unlimited cryptography (see page 32), an error message also appears.
No, use existing file	Enter the location of the host configuration file (SmHost.conf) in the text box, or click Choose to browse for the file. The default location of SmHost.conf is either: *ASA_HOME*\conf\ (Windows) or *ASA_HOME*/conf/ (UNIX)

12. In the Agent Configuration dialog, specify the name of the Agent Configuration Object that you created in the Policy Server User Interface before installing the SiteMinder Agent. Click Next.
13. In the Install Complete dialog, click Done to exit the installer.

    The installation is complete.

Running the Installation in Console Mode on UNIX

To install the SiteMinder Agent for WebSphere by running the installation script in a UNIX console:

1. Start the SiteMinder Policy Server, if it is not already running–the installation program connects to the Policy Server to create a trusted host.
2. Close all other programs.
3. Connect to the system where WebSphere is installed as the user who installed WebSphere. For example, if you installed as root, connect as root.
4. Download the following installation file to a temporary location:

   ca-asa-6.0-was-unix.bin
5. Depending on your permissions, you might need to add executable permissions to the installation file. For example:

   ```
   chmod +x ca-asa-6.0-was-sol.bin
   ```
6. Make sure the Java Virtual Machine (JVM) is specified in the PATH variable (see page 33).
7. In a Solaris, HP-UX, or Linux shell, in the temporary location, enter the following command:

   ```
   sh ./ca-asa-6.0-was-unix.bin -i console
   ```

   The -i console option interactively runs the installation from a console.
8. Read the License Agreement. If you accept the terms, enter Y and then press Enter.
9. In the Choose Install Folder section, specify a location for the SiteMinder Agent for IBM WebSphere installation, and then press Enter.

   CA recommends the following location:

   /opt/smwasasa

10. Enter **Y**, then press Enter to create or confirm the installation location for the SiteMinder Agent.

    The program installs the required files in the SiteMinder Agent install location.

11. Specify the installation location of the WebSphere Application Server. For example:

    /opt/WebSphere/AppServer

    The program installs the required files in the WebSphere install location.

12. At the Host Registration prompt, enter one of the following numbers:

    1 The installer invokes the Host Registration tool, smreghost, to register the unique trusted host name with the Policy Server and create the SmHost.conf file. Registering the system as a trusted host enables the SiteMinder Agent to establish a secure, trusted connection with the Policy Server.

    Before registering a trusted host, you must create a Host Configuration Object (see page 28) in the Policy Server.

    2 The installer invokes the smreghost tool to use an existing SmHost.conf file to establish the connection between the trusted host and the Policy Server.

    Note: Specify this option *only* if you are the reinstalling the SiteMinder Agent for WebSphere and the SmHost.conf file that you want to use was therefore created by the smreghost tool supplied with this release. The SiteMinder Agent for WebSphere is implemented using a pure Java SiteMinder Agent API and cannot use an SmHost.conf file created for another SiteMinder Agent to establish its connection to the Policy Server.

13. Do one of the following, then click Next:

If you entered...	Then do the following...
1 (to create a new trusted host)	Enter the following information: ▪ Policy Server IP Address–IP address of the Policy Server where you are registering the host ▪ SM Admin Username–Name of the administrator permitted to register the host with the Policy Server ▪ SM Admin Password–Password for the SM Admin account ▪ Host Name–Unique name that represents the trusted host to the Policy Server. The name does not have to be the same as the physical client system you are registering; it can be any unique name. ▪ Host Config Object–Name of the Host Configuration Object specified in the Policy Server. The installation program registers your unique trusted host name with the Policy Server. If your Policy Server is not running, a message appears and you can register the trusted host manually later (see page 41). If you have not patched the JVM Java Cryptography Extension (JCE) package for unlimited cryptography (see page 32), an error message also appears.
2 (to use an existing trusted host)	Enter the location of the host configuration file (smhost.conf). The default location of the file is: *ASA_HOME*/conf/

14. Supply the name of the Agent Configuration Object that you created for the SiteMinder Agent.
15. At the installation complete prompt, press Enter to exit the installer. The installation of the SiteMinder Agent for IBM WebSphere is complete.

Installing a Web Agent for Advanced TAI Authentication

If you are configuring the SiteMinder TAI to challenge requests for credentials, a SiteMinder Web Agent is required to collect credentials and authenticate user requests for authentication schemes other than Basic (the TAI can handle basic authentication itself).

If no suitable Web Agent is present in your SiteMinder environment, install and configure one (together with a supported Web Server).

For information about how to install and configure SiteMinder Web Agents, see the *CA eTrust SiteMinder Web Agent Installation Guide* and the *CA eTrust SiteMinder Agent Guide*.

More information:

SiteMinder Trust Association Interceptor (TAI) (see page 13)

Reregistering a Trusted Host Using the Registration Tool

When you install a SiteMinder Agent on a server for the first time, you are prompted to register that server as a trusted host. Once the trusted host is registered, you do not have to re-register with subsequent Agent installations.

There may be situations when you want to reregister a trusted host independent of an Agent installation, such as:

- To rename the trusted host if there has been a change to your SiteMinder environment
- To re-establish a trusted host if the trusted host has been deleted in the Policy Server User Interface
- To recrete policy objects if the trusted host policy objects have been deleted from the policy store or the policy store has been lost
- To change the shared secret that secures the connection between the trusted host and the Policy Server
- To recreate the SmHost.conf configuration file if it is lost
- To overwrite an existing trusted host without deleting it first

Reregistering a Trusted Host on Windows

To reregister a trusted host on Windows, use the Registration Tool, smreghost. This tool is installed when you install an Agent on a trusted host, and is located in the directory *ASA_HOME*\bin.

Note: When you re-register a host using smreghost, you must first remove the host from the Policy Server User Interface unless you use the smreghost command argument, **-o**, which lets you overwrite an existing trusted host without having to delete it from the Policy Server.

To run smreghost, enter the smreghost command using the following required arguments:

smreghost -i *policy_server_IP_address:port*
-u *administrator_username* -p *Administrator_password*
-hn *hostname_for_registration* -hc *host_configuration_ object*

Note: There should be a space between each command argument and its value.

Example:

smreghost -i 123.123.1.1 -u SiteMinder -p mypw -hn hostA
-hc DefaultHostSettings

Example with the -o argument:

smreghost -i 123.123.1.1 -u SiteMinder -p mypw -hn hostA
-hc DefaultHostSettings -o

More information:

smreghost Command Arguments (see page 44)

Reregistering a Trusted Host on UNIX

To reregister a trusted host on UNIX use the Registration Tool, smreghost. This tool is installed when you install an Agent on a trusted host, and is located in the directory *ASA_HOME*/bin.

Note: When you reregister a host using smreghost, you must first remove the host from the Policy Server User Interface unless you use the smreghost command argument, **-o**, which lets you overwrite an existing trusted host without having to delete it from the Policy Server.

To run smreghost:

1. Open a command prompt window.
2. Ensure that the library path environment variable contains the path to the SiteMinder Agent's bin directory.

   Enter the following two commands:

   library_path_variable=${*library_path_variable*}:*WS_HOME*/bin

   export *library_path_variable*

   where *library_path_variable* is LD_LIBRARY_PATH for Solaris and Linux and is SHLIB_PATH for HP-UX.

   Example: setting the library path

   To set the library path for for Solaris systems, enter the following two commands:

   LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/smwasasa/bin

   export LD_LIBRARY_PATH

3. Enter the smreghost command using the following required arguments:

   smreghost -i *policy_server_IP_address*:*port*
   -u *administrator_username* -p *Administrator_password*
   -hn *hostname_for_registration* -hc *host_configuration_ object*

   Note: There should be a space between each command argument and its value.

   Example: registering a trusted host

   smreghost -i 123.123.1.1 -u SiteMinder -p mypw -hn hostA
   -hc DefaultHostSettings

smreghost Command Arguments

The following table contains a complete list of command arguments for the smreghost tool.

Arguments	Value
i *policy_server_IP_ address*:*port*	IP address of the Policy Server where you are registering this host. Specify the port of the authentication server only if you are not using the default, which is 44442. If you specify a port number, which can be a non-default port, that port is used for all three Policy Server servers (authentication, authorization, accounting), however, the unified server responds to any Agent request on any port. The policyserver entry in the SmHost.conf file will be: "112.11.2.5,5555,5555,5555"
u *administrator_username*	Name of the SiteMinder administrator with the rights to register a trusted host.
p *administrator_password*	Password for the Administrator permitted to register a trusted host.
hn *hostname_for_registration*	Name of the host to be registered. This can be any name that identifies the host, but it must be unique. After registration, this name is placed in the Trusted Host list in the Policy Server User Interface.
hc *host_config_object*	Name of the Host Configuration Object configured at the Policy Server.

Arguments	Value
f *path_to_host_config_file*	Full path to the file that contains the registration data. The default file is SmHost.conf. If you do not specify a path, the file is installed in the location where you are running the smreghost tool. (Optional) If you use the same name as an existing host configuration file, the tool backs up the original and adds a .bk extension to the backup file name. On Windows systems, if you specify a file path with spaces, the entire path must be enclosed in quote marks.
cp *cryptographic_provider*	(Optional) Name of the cryptographic provider you are using for encryption. If you do not specify a value, BSAFE is the default.
cd *crypto_provider_DLL_path*	Full path to the PKCS11 DLL. This DLL is installed with the nCipher software installed on same Web server as the Web Agent. Required for PKCS11 encryption.
ct *crypto_provider_token_label*	Token label for the hardware token. Only use this argument if there is a token label. Optional for PKCS11 encryption.
ck *crypto_provider_token_pin*	Passphrase for the token. Required for PKCS11 encryption.
o *overwrite_existing_trusted_host*	Overwrites an existing trusted host without having to delete it first. The proper syntax is **-o**.

Reinstalling the SiteMinder Agent

If at any time you need to reinstall the SiteMinder Agent, first uninstall it and then install it.

More information:

Uninstalling the SiteMinder Agent (see page 46)
Installing the SiteMinder Agent for IBM WebSphere (see page 34)

Uninstalling the SiteMinder Agent

To uninstall SiteMinder Agent for IBM WebSphere, follow the procedures in this section.

Uninstalling from Windows

To uninstall SiteMinder Agent for IBM WebSphere from Windows:

1. Stop the WebSphere Server. The SiteMinder Agent cannot uninstall if WebSphere is running.
2. Navigate to *ASA_HOME*, where the SiteMinder Agent is installed.
3. Double-click asa-was-uninstall(.cmd).
4. In the Uninstall dialog, click Uninstall.
5. After confirmation indicates the uninstall is complete, click Done to exit.
6. If the uninstaller lists files it was not able to remove, you can manually remove them.
7. Manually remove the *ASA_HOME* directory (for example, smwasasa) that the installation created.

Uninstalling from UNIX

To uninstall SiteMinder Agent from UNIX platforms:

1. Stop the WebSphere Server. The SiteMinder Agent does not uninstall if WebSphere continues to run.
2. Make sure the PATH variable is set to the location of your JVM (see page 33).
3. Open a UNIX shell and navigate to *ASA_HOME*.
4. Enter the following command and press Enter to start the uninstall:
   sh ./asa-was-uninstall.sh
5. Remove the *ASA_HOME* directory (for example, smwasasa) that the installation created:
   a. Navigate to the directory one level above where the SiteMinder Agent is installed. For example:
      /opt
   b. Enter the following command and press Enter:
      rm -rf *ASA_HOME*.

What to Do After Installing the SiteMinder Agent

After installing the SiteMinder Agent for IBM WebSphere:

- Configure the SiteMinder Agent to work with the SiteMinder Policy Server (see page 49).
- Configure the SiteMinder Agent to work with WebSphere (see page 71).
- Verify the Agent installation and configuration (see page 85).
- Configure Policies (see page 91), if necessary.
- Troubleshoot the configuration (see page 123), if necessary.

Chapter 4: Configuring the SiteMinder Agent, SiteMinder-Side

This chapter describes configure the SiteMinder Agent to work with the SiteMinder Policy Server.

Note: Although all Agent modules are installed when you run the Agent installation, you need only configure the modules that you require.

This section contains the following topics:

Copying and Editing the smagent.properties File

The smagent.properties file specifies:

- Options for the SiteMinder Agent for IBM WebSphere's default connection log (SmWasAsaDefault.log)
- The locations of the Agent configuration files for the SiteMinder Agent for WebSphere modules

Sample smagent.properties file

```
##################################################################
# SiteMinder Generic Application Server Agent Properties File
##################################################################

logfilename="c:\smwasasa\log\SmWaAsaDefault.log"
loglevel="5"
logappend="NO"
logfile="YES"
logconsole="NO"
smazconf="c:\smwasasa\conf\AsaAgent-az.conf"
smauthconf="c:\smwasasa\conf\AsaAgent-auth.conf"
smassertionconf="c:\smwasasa\conf\AsaAgent-assertion.conf"
```

More information:

Preconfiguring Policy Objects for the SiteMinder Agent (see page 27)
Fine-Tuning the Agent Configuration Setup (see page 52)
Logging (see page 107)

Copying the smagent.properties File to WebSphere

You must copy the smagent.properties file from the default location (ASA_*HOME*/conf/) to the following directory in your WebSphere Application Server installation:

WS_HOME\profiles*myDefaultProfileName*\properties\

Editing smagent.properties

Generally, you only need to edit the smagent.properties file to change SiteMinder Agent logging options (see page 107) or if you change the names or locations of your configured Agent configuration files. For example, if you are using a shared Agent configuration file (see page 56).

In clustered and SSO WebSphere environments, the smagent.properties file is replicated on many systems. However, the SiteMinder Agent may not be installed in the same file system location. The Agent configuration file locations specified in smagent.properties may not therefore be correct for all systems in such an environment.

To handle this situation, you can define a JVM system property, **smasa.home**, which defines the installed location of the SiteMinder Agent on the local host and then edit smagent.properties to remove absolute paths to the Agent configuration files. Where the absolute path is absent, the SiteMinder Agent uses the value of smasa.home to determine where to find the configuration files.

For example, change the first line to resemble the second line:

```
smazconf="c:\smwasasa\conf\AsaAgent-az.conf"

smazconf="AsaAgent-az.conf"
```

To set the smasa.home JVM system property (on each WebSphere server in the cluster or SSO environment):

1. Open the WebSphere administrative console.
2. Click the following, in the order shown:

   Servers, Application Server, server1, Java and Process Management, Process Definition, Java virtual Machine, Additional Properties, Custom Properties.
3. Create a new variable in Custom Properties named smasa.home and specify its value as *ASA_HOME*. For example, in Windows enter:
   ```
   smasa.home=C:\smwasasa
   ```
4. Save the changes in master configuration file and restart the server.
5. Check Systemout.log file for the server instance.

Fine-Tuning the Agent Configuration Setup

By default, the SiteMinder ASA installation creates an Agent configuration file for each Agent module:

Module	Agent Configuration File
SiteMinder TAI	AsaAgent-assertion.conf
SiteMinder Login Module	AsaAgent-auth.conf
SiteMinder JACC Provider	AsaAgent-az.conf

The Agent configuration files are located in the *ASA_HOME*\conf directory, where *ASA_HOME* is the location where you installed the ASA. For example:

- For Windows

  C:\smwasasa\conf

- For UNIX

  /opt/smwasasa/conf

Each Agent configuration file is created with the following default configuration parameters/values:

Parameter	Default Value
EnableWebAgent	Yes (the SiteMinder Agent is enabled by default)
HostConfigFile	Local Host Configuration File (typically *ASA_HOME*\conf\SmHost.conf or the location of the existing SmHost.conf file you specified during Trusted Host registration)
AgentConfigObject	The Agent Configuration Object specified during installation

After installation, each Agent module has its own configuration file and all three configuration files reference the same Agent Configuration Object and Agent identity. However, you can change this arrangement to suit your needs by doing one of the following:

- Creating separate Agent Configuration Objects for each module on the Policy Server and change the AgentConfigObject parameters in each Agent configuration file to reference the appropriate objects.
- Creating a single, shared Agent configuration file (for example, named AsaAgent.conf) for all three modules.

Note: For TAI-only configurations, create and configure a single Agent Configuration Object and configure the AsaAgent-assertion.conf file that references it.

The following table describes the features, benefits, and drawbacks of each possible Agent configuration arrangement:

Configuration	Features	Benefits/Drawbacks
Each Agent module has a separate Agent configuration file. All configuration files reference the same Agent Configuration Object. (*Default*)	▪ Module-specific Agent configuration parameters are defined locally in the Agent configuration files. ▪ Common Agent configuration parameters are defined centrally in the Agent Configuration Object on the Policy Server.	**Benefits:** ▪ Allows fine-grained configuration of cache settings. For example, you can configure an authorization cache size of 0 for the SiteMinder TAI and Login Modules (which do not perform authorization), but increase the cache size for the SiteMinder JACC Provider (which does). ▪ Allows Module-specific information to be written to separate log files. That is, you can configure separate log files for TAI messages, Login Module messages, and JACC Provider messages, increasing readability. ▪ Allows modules to be individually enabled/disabled. **Drawback:** ▪ Module-specific settings in local configuration files must be edited locally on each WebSphere host whenever a change is required.

Configuration	Features	Benefits/Drawbacks
Each Agent module has a separate Agent configuration file. Each Agent configuration file references a separate Agent Configuration Object.	▪ Agent configuration parameters for each module are defined centrally in separate Agent Configuration Objects on the Policy Server. ▪ Module-specific configuration is encapsulated in that module's object.	**Benefits:** ▪ Allows fine-grained configuration of cache settings. For example, you can configure an authorization cache size of 0 for the SiteMinder TAI and Login Modules (which do not perform authorization), but increase the cache size for the SiteMinder JACC Provider (which does). ▪ Allows Module-specific information to be written to separate log files. That is, you can configure separate log files for TAI messages, Login Module message, and JACC Provider messages. ▪ Agent configuration settings can be applied on multiple hosts and managed centrally from the Policy Server. **Drawback:** ▪ Separate configuration objects must be maintained for each module even though most parameter values are common.
All Agent modules share the same Agent configuration file and reference the same Agent Configuration Object. (*Not recommended*)	▪ Agent configuration parameters for all modules are defined centrally in the Agent Configuration Object on the Policy Server and applies to all modules.	**Benefits:** ▪ Simplest to maintain. **Drawbacks:** ▪ Cannot enable/disable individual modules. ▪ Hardest to troubleshoot; information from all modules is written to the same log file, decreasing readability. ▪ Does not allow fine-grained, module-specific configuration.

Note: When using separate Agent Configuration Objects/Agent identities for each module, ensure that the SiteMinder TAI and JACC Provider modules all authenticate/authorize against the same realms in the Policy Server. You can accomplish this by configuring them in an Agent group.

More information:

Preconfiguring Policy Objects for the SiteMinder Agent (see page 27)

Using One Agent Configuration Object and Multiple Agent Configuration Files

The SiteMinder TAI, SiteMinder Login Module, and SiteMinder JACC Provider have their own Agent configuration files that each reference the same Agent Configuration Object *by default*. You do not need to take any further steps to use this arrangement. However, you will need to define Agent configuration parameters, as required, for each module.

Using Module-Specific Agent Configuration Objects

By default, the SiteMinder TAI, SiteMinder Login Module, and SiteMinder JACC Provider each have their own Agent configuration files that all reference the same, previously created Agent Configuration Object. However, you can create separate Agent Configuration Objects for each module, enabling centralized control of settings for each module from the Policy Server.

To configure the SiteMinder Agent to use separate Agent Configuration Objects for each ASA module:

1. In the Policy Server User Interface, do the following for *each* Agent module type:
    a. Create a new Agent identity with a name appropriate for the module that it represents (for example, WSAgentTAI).
    b. Create a duplicate of the Agent Configuration Object that you created for the SiteMinder Agent components before installation.
    c. Set the DefaultAgentName parameter to the Agent identity defined in Step a. You can also set other module-specific Agent configuration parameters (see page 114), if you know them. Otherwise, these are described in-context later.
    d. Save the Agent Configuration Object with a name appropriate for the module to which it relates (for example, AsaTAISettings).

2. On the system where the SiteMinder Agent is installed:
   a. Edit the AsaAgent-assertion.conf file to change the value of the **AgentConfigObject** parameter to match the name of the Agent Configuration Object that you created for SiteMinder TAI modules.
   b. Edit the AsaAgent-auth.conf file to change the value of the **AgentConfigObject** parameter to match the name of the Agent Configuration Object that you created for SiteMinder Login Module modules.
   c. Edit the AsaAgent-az.conf file to change the value of the **AgentConfigObject** parameter to match the name of the Agent Configuration Object that you created for SiteMinder JACC Provider modules.

Note: The single Agent identity and Agent Configuration Object you created before installation (see page 28) should nos longer be in use; you can delete them now if you wish.

Using a Shared Agent Configuration File and Configuration Object for All Agent Modules

By default, the SiteMinder TAI, SiteMinder Login Module, and SiteMinder JACC Provider that comprise a SiteMinder Agent for IBM WebSphere each have their own Agent configuration file. However, you can configure all the Agent modules to share a single Agent Configuration file (and thus, a single configuration object).

To create a shared Agent configuration file and configuration object for all three modules of a SiteMinder Agent for IBM WebSphere, on the system where the SiteMinder Agent is installed:

1. Create a shared Agent configuration file by copying any one of the AsaAgent-*module*.conf files and giving it a new name (for example, AsaAgent.conf).
2. Open the shared Agent configuration file and ensure that the agentname, HostConfigFile, and AgentConfigObject parameters are configured correctly.
3. Edit the smagent.properties (see page 50) file to change the value of the smazconf, smauthconf, and smassertionconf parameters to all reflect the new shared Agent configuration file name. For example:

```
smazconf="c:\smwasasa\conf\AsaAgent.conf"
smauthconf="c:\smwasasa\conf\AsaAgent.conf"
smassertionconf="c:\smwasasa\conf\AsaAgent.conf"
```

Note: Because of the limitations associated (see page 52) with this configuration, it is not generally recommended.

Configuring the TAI, SiteMinder-Side

This section describes how to perform SiteMinder-side configuration of the SiteMinder TAI (that is, configuring the SiteMinder TAI to work with the SiteMinder Policy Server).

More information:

SiteMinder Trust Association Interceptor (TAI) (see page 13)
Configuring the SiteMinder TAI in WebSphere (see page 75)

Configuring the TAI to Only Handle Requests from SiteMinder Session Holders

To configure the SiteMinder TAI to only handle requests from users with SiteMinder session tickets:

- Make sure the ChallengeForCredentials Agent configuration parameter is not set (see page 57).
- Set the AssertionAuthResource Agent configuration parameter (see page 57).
- Create an Assertion realm for non-challenged requests (see page 58).

Disabling the ChallengeForCredentials Agent Configuration Parameter

To configure the SiteMinder TAI to handle only requests from users with an existing SiteMinder session ticket (that is, to not challenge requests for credentials), you must ensure that the **ChallengeForCredentials** Agent configuration parameter is disabled by removing it (the default is NO) or setting it to NO in the associated Agent Configuration Object or Agent configuration file.

For example:

```
ChallengeforCredentials=NO
```

Setting the AssertionAuthResource Agent Configuration Parameter

If you are configuring the TAI to *not* challenge requests for credentials, you must define the **AssertionAuthResource** Agent configuration parameter in the associated Agent Configuration Object or Agent configuration file.

The value of AssertionAuthResource *must* match the value specified for the resource filter in the assertion realm that you create for non-challenged requests.

Note: In earlier SiteMinder TAI implementations, the assertion realm was referred to as a *validation* realm and had a static resource filter (/siteminderтai). If you have an existing validation realm, you do not need to change it. However, you must set the AssertionAuthResource Agent configuration parameter to refer to it.

For example:

```
assertionauthresource=/siteminderassertion
```

Creating an Assertion Realm for Non-Challenged Requests

If your SiteMinder TAI is not configured to challenge requests for credentials (the challengeforcredentials Agent configuration parameter is set to **no**), you configure a *SiteMinder TAI Assertion Realm* in which SiteMinder simply asserts the identities obtained from SiteMinder session cookies associated with HTTP requests. This assures that requests by HTTP clients already authenticated by SiteMinder (and thus with associated SiteMinder session cookies) are not re-challenged by WebSphere when they access your Web applications. Other requests are rejected.

Note: In earlier SiteMinder TAI implementations, this realm was referred to as a *validation* realm and had a static resource filter (/siteminderтai). If you have an existing validation realm, you do not need to change it. However, you must set the AssertionAuthResource Agent configuration parameter to refer to it.

To create a realm for non-challenged requests:

1. Start the SiteMinder Policy Server User Interface.
2. On the System tab, right-click User Directories and select Create User Directory to make a user directory configured to the same LDAP user store as the one used by WebSphere. For more information on creating user directories, see *CA eTrust SiteMinder Policy Design*.
3. Right-click Domains and select Create Domain to create a policy domain that you want to protect. Assign the user directory from Step 2 to this domain. For more information on creating domains, see *CA eTrust SiteMinder Policy Design*.
4. On the Domains tab, right-click the domain and select Create Realm.
5. On the Resource tab of the SiteMinder Realm dialog, specify properties:

Name:	SiteMinder TAI Assertion Realm
Description:	SiteMinder TAI Assertion Realm
Agent:	The SiteMinder Agent Identity you configured for the SiteMinder TAI

Resource Filter:	*/AssertionAuthResource* (any value is valid, but it must match value of AssertionAuthResource Agent configuration parameter specified for the TAI module) For example, /siteminderassertion
Authentication Scheme:	Basic or any authentication scheme

6. Click the Session tab.
7. Disable any session time-outs and make sure the No Persistent Session option is enabled.
8. Set the Default Resource Protection for the realm to Protected.
9. Click Apply and OK.

Configuring rules or policies for this assertion realm is unnecessary. However, to implement user mapping, you must set an authentication response attribute, and then configure appropriate rules and policies for the assertion realm (see page 97).

Configuring the TAI to Challenge Requests for Credentials

To configure the SiteMinder TAI to challenge requests from users without an existing SiteMinder session ticket as well as handle those that do have an existing SiteMinder session:

- Set the ChallengeForCredentials Agent configuration parameter (see page 59)
- If using advanced authentication, synchronize overlapping settings (see page 60) between the TAI and the Web Agent performing authentication
- If using advanced authentication, configure the authentication scheme you want to use (see page 61)

Setting the ChallengeForCredentials Agent Configuration Parameter

To configure the SiteMinder TAI to challenge requests from users without an existing SiteMinder session ticket as well as handle those that do have an existing SiteMinder session, you must set the **ChallengeForCredentials** Agent configuration parameter to "YES" in the associated Agent Configuration Object or Agent configuration file.

For example:

```
ChallengeforCredentials=YES
```

Default is NO.

Synchronizing Overlapping SiteMinder TAI and Web Agent Configuration Parameters

When configured to challenge requests for credentials, for authentication schemes other than basic, the SiteMinder TAI module redirects to a Web Agent to collect credentials. Because of this, you must ensure that several Agent configuration parameters that apply to both Agent types have matching values.

The fcccompatmode Agent configuration parameter handles backward compatibility of forms credential collection, which the SiteMinder TAI does not support. You must therefore set this parameter to NO for both the SiteMinder TAI and the Web Agent:

```
fcccompatmode="NO"
```

The SiteMinder TAI does not support legacy encoding. Set the legacyencoding Agent configuration parameter to NO for both the SiteMinder TAI and the Web Agent:

```
legacyencoding="NO"
```

The secureURLs setting in the Agent Configuration Object does not affect the fcccompatmode and legacyencoding parameters – theSiteMinder TAI does not support them no matter what secureURLs is set to.

Note: The secureURLs parameter enables the Web Agent to encrypt all Siteminder query parameters in a redirection URL. When this parameter is set to yes, the Agents will encrypt query data when it returns an HTTP 302 status code (redirect response) to the browser. This functionality can be used when a requested resource is protected by an advanced authentication scheme. Use the Policy Server User Interface to centrally set SecureURLs in the Agent Configuration Object.

Additionally, the following parameters must match for both the SiteMinder TAI and SiteMinder Web Agent if specified:

- EncryptAgentName
- IgnoreQueryData

Note: Some configuration parameter values must also match for the SiteMinder JACC Provider, if configured. A complete list of Agent configuration parameters with interdependencies noted for all modules is included in Agent Configuration Parameters (see page 114).

More information:

TAI-Specific Agent Configuration Parameter Summary (see page 62)
Agent Configuration Parameters (see page 114)

Configuring an Authentication Scheme for Challenged Requests

If you are configuring the SiteMinder TAI to challenge requests for credentials using non-Basic authentication, you must configure the required authentication scheme, if it does not exist already.

For more information, see *CA eTrust SiteMinder Policy Design*.

Creating Realms for Challenged Requests

If your SiteMinder TAI is configured to challenge HTTP requests for credentials (the challengeforcredentials Agent configuration parameter is set to **yes**), you configure standard SiteMinder protection domains and realms to protect your Web container resources.

If you are also configuring the SiteMinder JACC Provider, you do not need to create realms for challenged requests now, but can do so later as part of the policy configuration process (see page 91).

If you are configuring a TAI-only environment, you should familiarize yourself with SiteMinder resource mapping conventions for Web applications (see page 93). In general, realms to protect your Web applications should have properties similar to these:

Name:	Example Web App Protection Realm.
Description:	SiteMinder realm for validating/authenticating identities using the TAI.
Agent:	The Agent identity associated with the SiteMinder TAI.
Resource Filter:	*/web_app_context* Where *web_app_context* is the J2EE Web application context for the protected Web application. For example, /mywebapp. SiteMinder Resource Mapping for WebSphere Resources (see page 93).
Authentication Scheme:	The authentication scheme to use to collect credentials from and authenticate user requests. The SiteMinder TAI handles Basic authentication itself; other authentication schemes must be processed by a Web Agent.

To implement user mapping, you must set an authentication response attribute, and then configure appropriate policies for the assertion realm (see page 97).

More information:

Configuring Policies for the SiteMinder Agent (see page 91)
Resource Mapping (see page 93)
Configuring SiteMinder Policies to Support User Mapping (Optional) (see page 97)

TAI-Specific Agent Configuration Parameter Summary

Define the following Agent configuration parameters for the SiteMinder TAI in an associated Agent Configuration Object or Agent configuration file.

Required Parameter	Value	Description
AcceptTpCookie	**yes** or **no**	Configures the SiteMinder TAI to assert identities from third-party SiteMinder session cookies generated using the SiteMinder SDK. For details, see "Enabling Single Sign-On" in the Agent API chapter of: ▪ CA eTrust SiteMinder Developer's Guide for C ▪ CA eTrust SiteMinder Developer's Guide for Java Default is NO. **Note:** If you configure the SiteMinder TAI to accept third-party SiteMinder session cookies, you must also configure the SiteMinder Login Module to accept them so that it can assert WebSphere propagation tokens in situations when WebSphere must reestablish Subjects created by the SiteMinder TAI.
ChallengeForCredentials	**yes** or **no**	Specifies whether the SiteMinder TAI should challenge for credentials. Default is NO.
AssertionAuthResource	String	If you are configuring the TAI to *not* challenge requests for credentials, this value *must* match the value specified for the resource filter in the realm that you create for non-challenged requests (see page 58). For example: assertionauthresource=/siteminderta i

Required Parameter	Value	Description
CookieDomain	String	Name of the cookie domain. For example: cookiedomain="ca.com" No default value. See also the cookiedomainscope parameter.
CookieDomainScope	Number	If specified, further defines the cookie domain for assertion of SiteMinder session cookies by the SiteMinder TAI. The scope determines the number of sections, separated by periods, that make up the domain name. A domain always begins with a period (.) character. For example: cookiedomainscope="2" Default is 0, which takes the domain name specified in the cookiedomain parameter.
EncryptAgentName	**yes** or **no**	Specifies whether the agent name should be encrypted when redirecting to the SiteMinder Web Agent for SiteMinder TAI credential collection. Default is NO.
FccCompatMode	**yes** or **no**	Specifies whether to handle backward compatibility of forms credential collection, which the SiteMinder TAI does not support. You must therefore set this parameter to NO for *both* the SiteMinder TAI *and* the Web Agent: fcccompatmode="NO"
ServerErrorFile	String	Specifies a page to redirect a request to if a processing error is encountered. This can either be an HTTP or local file system resource. For example: servererrorfile="http://server.ca.com:88/errorpage.html" If this setting is not configured, a default message is output to the response when the TAI encounters an error. The default message is "SiteMinder Agent encountered an error while handling request. Please ask the administrator to look for messages in the server's agent log to check for the cause."

More information:

Agent Configuration Parameters (see page 114)

What to Do Next if You Are Setting Up a TAI-Only Configuration

If you are setting up a TAI-only SiteMinder Agent configuration, skip the rest of the procedures in this chapter and proceed to Configuring the SiteMinder Agent, WebSphere-side (see page 71).

Configuring the Login Module, SiteMinder-Side

This section describes how to perform SiteMinder-side configuration of the SiteMinder Login Module (that is, configuring the SiteMinder Login Module to work with the SiteMinder Policy Server).

More information:

SiteMinder Login Module (see page 15)
Configuring the Login Module in WebSphere (see page 77)

Configuring the Login Module to Handle Java Client Requests

To configure the SiteMinder Login Module to handle Java client (RMI-IIOP) requests for EJB container resources in SiteMinder, you must configure:

- The RmiAuthResource Agent configuration parameter (see page 64)
- An RMI realm (see page 65)

Setting the RmiAuthResource Agent Configuration Parameter

To configure the SiteMinder Login Module to handle Java client (RMI-IIOP) requests, you must define the **RmiAuthResource** Agent configuration parameter in the associated Agent Configuration Object or Agent configuration file.

The value of RmiAuthResource is a string that must match the value you specify for the resource filter in the realm that you create for Java Client requests (see page 65).

For example:

```
RmiAuthResource=/siteminderrmirealm
```

More information:

Creating a Realm for Java Client (RMI) Requests (see page 65)

Creating a Realm for Java Client (RMI) Requests

Create a realm in which the Login Module authenticates identities associated with Java client (RMI) requests for EJB container resources.

To create a realm for Java Client requests:

1. Start the SiteMinder Policy Server User Interface.
2. On the System tab, right-click User Directories and select Create User Directory to make a user directory configured to the same LDAP user store as the one used by WebSphere. For more information on creating user directories, see *CA eTrust SiteMinder Policy Design*.
3. Right-click Domains and select Create Domain to create a policy domain that you want to protect. Assign the user directory from Step 2 to this domain. For more information on creating domains, see *CA eTrust SiteMinder Policy Design*.
4. On the Domains tab, right-click the domain and select Create Realm.
5. On the Resource tab of the SiteMinder Realm dialog, specify properties:

Name:	SiteMinder RMI Realm
Description:	SiteMinder Login Module Java Client (RMI) Assertion Realm
Agent:	The SiteMinder Agent Identity you configured for the SiteMinder Agent for IBM WebSphere
Resource Filter:	*/smrmirealm* (any value is valid, but it must match the value of the RmiAuthResource Agent configuration parameter that you specify for the Login Module) For example, /siteminderrmirealm
Authentication Scheme:	Basic or any authentication scheme

6. Click the Session tab.
7. Set applicable session timeouts.

8. Make sure the No Persistent Session radio button is enabled.
9. Click Apply and OK.

Configuring rules or policies for the RMI realm is generally unnecessary. However, to implement user mapping, you must set an authentication response attribute, and then configure appropriate rules and policies for the RMI realm.

More information:

Setting the RmiAuthResource Agent Configuration Parameter (see page 64)
Configuring SiteMinder Policies to Support User Mapping (Optional) (see page 97)

Configuring the Login Module to Handle System Login Requests

To configure the SiteMinder Login Module to handle System Login (J2EE RunAs Identity) requests for EJB container resources, you must configure:

- SystemAuthResource Agent configuration parameter (see page 66)
- System Login realm (see page 66)

Setting the SystemAuthResource Agent Configuration Parameter

To configure the SiteMinder Login Module to handle System Login requests in SiteMinder, you must define the **SystemAuthResource** Agent configuration parameter in the associated Agent Configuration Object or Agent configuration file.

The value of SystemAuthResource is a string that must match the value you specify for the resource filter in the realm that you create for System Login requests (see page 66).

For example:

```
SystemAuthResource=/siteminderssystemrealm
```

More information:

Fine-Tuning the Agent Configuration Setup (see page 52)
Creating a Realm for System Login (J2EE RunAs Identity) Requests (see page 66)

Creating a Realm for System Login (J2EE RunAs Identity) Requests

You must create a realm in which the Login Module authenticates identities associated with System Login requests for EJB container resources.

To create a realm for non-challenged requests:

1. Start the SiteMinder Policy Server User Interface.
2. On the System tab, right-click User Directories and select Create User Directory to make a user directory configured to the same LDAP user store as the one used by WebSphere. For more information on creating user directories, see *CA eTrust SiteMinder Policy Design*.
3. Right-click Domains and select Create Domain to create a policy domain that you want to protect. Assign the user directory from Step 2 to this domain. For more information on creating domains, see *CA eTrust SiteMinder Policy Design*.
4. On the Domains tab, right-click the domain and select Create Realm.
5. On the Resource tab of the SiteMinder Realm dialog, specify properties:

Name:	SiteMinder System Login Realm
Description:	SiteMinder Login Module System Login Assertion Realm
Agent:	The SiteMinder Agent Identity you configured for the SiteMinder Agent for IBM WebSphere
Resource Filter:	/*smsystemrealm* (any value is valid, but it must match value of SystemAuthResource Agent configuration parameter specified for the Login Module). For example, /sitemindersystemrealm
Authentication Scheme:	Basic or any authentication scheme

6. Click the Session tab.
7. Set applicable session timeouts (greater than the value specified for the WebSphere cache timeouts which apply to the WebSphere created RunAs Subject).

8. Make sure the No Persistent Session radio button is enabled.
9. Click Apply and OK.

Configuring rules or policies for the System Login realm is generally unnecessary. However, to implement user mapping, you must set an authentication response attribute, and then configure appropriate rules and policies for the System Login realm.

More information:

Setting the SystemAuthResource Agent Configuration Parameter (see page 66)
Configuring SiteMinder Policies to Support User Mapping (Optional) (see page 97)

Login Module-Specific Agent Configuration Parameter Summary

Define the following Agent configuration parameters in the appropriate associated Agent Configuration Object or Agent configuration file. Setting Up Agent Configuration Files and Objects (see page 52)

Required Parameter	Value	Description
AcceptTpCookie	**yes** or **no**	If you configured the SiteMinder TAI to accept third-party SiteMinder session cookies (see page 62), configure this parameter for the SiteMinder Login Module so that it can assert WebSphere propagation tokens in situations when WebSphere must reestablish Subjects created by the SiteMinder TAI. For example: AcceptTpCookie=Yes Default is NO.
RmiAuthResource (Required to support Java client requests)	String	Specifies the resource used when authenticating Java client (RMI-IIOP) requests. This value must match the value specified for the resource filter in the realm that you create for Java Client requests. For example: RmiAuthResource=/smrmirealm

Required Parameter	Value	Description
SystemAuthResource (Required to support System login requests)	String	Specifies the resource used when authenticating System login (J2EE RunAs identity) requests. This value must match the value specified for the resource filter in the realm that you create for System Login requests. For example: SystemAuthResource=/smsystemrealm

More information:

TAI-Specific Agent Configuration Parameter Summary (see page 62)
Agent Configuration Parameters (see page 114)

Configuring the SiteMinder JACC Provider, SiteMinder-Side

This section describes how to perform SiteMinder-side configuration of the SiteMinder JACC Provider (that is, configuring the SiteMinder JACC Provider to work with the SiteMinder Policy Server).

More information:

SiteMinder Java Authorization Contract for Containers (JACC) Provider (see page 17)
Configuring the SiteMinder JACC Provider in WebSphere (see page 81)
Configuring Policies for the SiteMinder Agent (see page 91)

Configuring Policies for the SiteMinder JACC Provider

If you are using the SiteMinder JACC, you configure standard SiteMinder protection domains, realms, and authorization policies to protect your WebSphere resources.

More information:

Configuring Policies for the SiteMinder Agent (see page 91)

JACC-Specific Agent Configuration Parameters

Define the following JACC Provider-specific Agent configuration parameters in the appropriate associated Agent Configuration Object or Agent configuration file as needed (there are no required parameters).

Parameter	Value	Description
AzCacheSize	Number	Size of the authorization cache (in number of entries) for the JACC Provider. For example: authcachesize="1000" Default is 0. To flush this cache, use the Policy Server User Interface.
IgnoreExt	Comma separated string	Species common file extensions (.gif, .jpg, .jpeg, .png, and .class) that the JACC Provider can ignore. The JACC Provider passes requests for files with these extensions directly to WebSphere without authorization. Use this parameter to specify extensions of files that do not require as much security as other resources

Note: Some configuration parameter values must also match those configured for the SiteMinder TAI. A complete list of Agent configuration parameters with interdependencies noted for all modules is included in Agent Configuration Parameters (see page 114).

All other SiteMinder-side SiteMinder JACC Provider configuration is covered in Configuring Policies for the SiteMinder Agent (see page 91).

What to Do After Completing SiteMinder-Side Configuration

After completing SiteMinder-side configuration of the SiteMinder Agent modules:

1. Perform WebSphere-side configuration of the SiteMinder Agent for IBM WebSphere (see page 71).
2. Verify that your SiteMinder Agent is working correctly (see page 85).
3. Configure SiteMinder authorization policies (see page 91), if necessary.
4. Troubleshoot the configuration (see page 121), if necessary.

Chapter 5: Configuring the SiteMinder Agent, WebSphere-Side

This chapter describes WebSphere-side configuration of the SiteMinder Agent for IBM WebSphere (that is, configuring the SiteMinder Agent to work in the WebSphere Application Server).

Note: Although all Agent modules are installed when you run the Agent installation, you need only configure the modules that you require. Information on which components to configure for your environment can be found in Choosing the Agent Modules You Need (see page 21).

This section contains the following topics:

Configuring General WebSphere Settings

Before you configure the SiteMinder Agent modules in your deployment:

- Configure the active user registry for security (see page 72)
- Enable WebSphere Global Security (see page 73)
- Enable Security Attribute Propagation for WebSphere SSO (see page 74), if required

Configuring LDAP as a WebSphere User Registry

In a typical deployment, WebSphere and the SiteMinder Policy Server are configured to use the same LDAP user registry.

Note: If you are not configuring WebSphere and the Policy Server to use the same LDAP user registry (typically because WebSphere is already configured with a custom user registry), ensure that the custom registry is properly configured (see the WebSphere documentation for information) and configure user mapping.

To configure a SiteMinder LDAP user directory as a WebSphere user registry:

1. If necessary, start the WebSphere Server and the WebSphere Administrative Console.
2. In the WebSphere Administrative Console, select Security, Global Security.
3. From the Active user registry drop-down menu, select Lightweight Directory Access Protocol (LDAP).
4. Click Apply.
5. Under User registries, click LDAP.
6. Under General Properties on the LDAP User Registry page, fill in the following fields and then click Apply.
   - Server user ID
   - Server user Password
   - Type
   - Host
   - Port
   - Base Distinguished Name (DN)
   - Bind Distinguished Name (DN)
   - Bind Password
   - Search timeout

7. Depending on the WebSphere configuration, check Reuse Connection and Ignore case for authorization.
8. Click Apply to apply your changes. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

Identity and User Mapping (see page 18)
Configuring SiteMinder Policies to Support User Mapping (Optional) (see page 97)

Enabling WebSphere Global Security

To enable global security for the WebSphere managed domain:

1. If necessary, start the WebSphere Server and the WebSphere Administrative Console.
2. In the WebSphere Administrative Console, select Security, Global Security.
3. From the Active authentication mechanism drop-down menu, select Lightweight Third Party Authentication (LTPA).
4. Set the Enable global security option.
5. Set the Enforce Java 2 security option.
6. Set the Cache timeout to a lesser value than the SiteMinder max time-out value for any associated SiteMinder realm that the user might first enter in WebSphere.

   Note: If the SiteMinder Login Module will be configured to handle system login (J2EE runAs) requests (see page 78), the Cache timeout must be set to a value less than 3 times the SiteMinder Agent key rollover period defined on the Policy Server. Thus, if Agent key rollover period is one hour, the Cache timeout value should be less than 180 minutes.
7. Click Apply to apply your changes. To save changes, click System Administration and Save Changes to Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

Configuring the Login Module to Handle System Login Requests (see page 66)
Session Handling (see page 105)

Enabling Security Attribute Propagation for WebSphere SSO

If WebSphere Single Signon (SSO) is configured, you must also enable Web inbound security attribute propagation to ensure that LTPA tokens are used.

To enable security attribute propagation:

1. If necessary, start the WebSphere Server and the WebSphere Administrative Console.
2. In the WebSphere Administrative Console, select Security, Global Security.
3. Under Authentication mechanisms, click LTPA.
4. Under Additional properties, click Single Signon (SSO).
5. Under General properties, set the Web inbound security attribute propagation option.
6. Click Apply to apply your changes. To save changes, click System Administration and Save Changes to Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

Configuring the Class Loader for the SiteMinder Agent Logger

The SiteMinder Agent for IBM WebSphere Logger is implemented using Apache's log4j. (see http://logging.apache.org/log4j/docs/). The log4j software is therefore packaged and installed with the SiteMinder Agent. The SiteMinder Agent Logger is packaged in WS_HOME/lib/ext/smlogger.jar and also uses the Apache log4j.jar located in the same directory.

Because log4j is an open source component, other J2EE applications deployed in WebSphere may also use it to implement logging. J2EE applications achieve isolation between different log4j versions by providing log4j in the application classpath. However, unless you configure a class loader, system components like the SiteMinder Agent would otherwise require log4j to be present in the system classpath (which would cause accompanying issues).

The SiteMinder Agent class loader enables the SiteMinder Agent Logger to be loaded outside of the container system classpath so that log4j does not need to be placed in the system classpath. The SiteMinder Agent then loads log4j and the dependent SiteMinder Agent logger classes from another location.

To configure the SiteMinder Agent class loader in WebSphere:

1. Move the smlogger.jar and log4j.jar files from *WS_HOME*/lib/ext to *ASA_HOME*/lib.
2. If you have not already done so, set the Java system environment variable smasa.home (see page 51) to point to *ASA_HOME* (for example, smasa.home=c:\smwasasa).
3. Set the Java system environment variable log4j.ignoreTCL to true (that is, log4j.ignoreTCL=true).
4. Grant J2SE permissions to the jar files under *ASA_HOME*/lib by adding them to the server.policy file in *WS_HOME*/profiles/*My_Profile_Name*/properties:

```
grant codeBase "file:/ASA_HOME/lib/-" {
  permission java.security.AllPermission;
};
```

When the WebSphere Application Server is started, the SiteMinder Agent should now detect that the logger class is not available in the system classpath and will try and load the logger from *smasa.home*/lib (that is, the location in which the smlogger.jar and log4j.jar files are placed).

More information:

Editing smagent.properties (see page 51)
Logging (see page 107)

Configuring the SiteMinder TAI in WebSphere

You configure the SiteMinder TAI in the WebSphere Application Server using the WebSphere Administrative Console. General information about enabling Web Trust Associations is available in the WebSphere documentation.

Note: If you are upgrading from an earlier SiteMinder TAI implementation and therefore have the earlier SiteMinder TAI configured in WebSphere, you must change the Interceptor classname SiteMinderTrustAssociationInterceptor to reflect the new TAI implementation. You should also remove the siteminder_tai_conf custom property (the location of the Agent configuration files are now specified in the smagent.properties file). The SiteMinderTrustAssociationInterceptor class and siteminder_tai_conf custom property are deprecated at this release.

To enable the Web Trust Association between the SiteMinder TAI and the WebSphere Application Server:

1. If necessary, start the WebSphere Server and the WebSphere Administrative Console.
2. Select Security, Global Security.
3. Under Authentication, select Authentication Mechanisms, LTPA.
4. Under Additional Properties, select Trust association.
5. Under General Properties on the Trust association page, set the Enable Trust Association option.
6. Under Additional Properties, click Interceptors.
7. On the Interceptors page, click New.
8. Under General Properties on the New page, enter the following SiteMinder TAI classname next to Interceptor Classname and click Apply:

   com.netegrity.siteminder.websphere.auth.SmTrustAssociationInterceptor

9. Click Apply to save your changes. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

Note: WebSphere supports the co-existence of multiple TAI module implementations. The TAI implementation used is determined as follows: when a request is made for a WebSphere protected resource, the server will call each TAI implementation's isTargetInterceptor () method one by one to determine which interceptor is going to handle the request until one responds.

TAI-Only Configurations:

If you are setting up a SiteMinder TAI-only configuration, skip the rest of the procedures in this chapter and proceed to Verifying SiteMinder Agent Installation and Configuration (see page 85).

More information:

SiteMinder Login Module (see page 15)
Configuring the TAI, SiteMinder-Side (see page 57)
Verifying SiteMinder Agent Installation and Configuration (see page 85)

Configuring the Login Module in WebSphere

You configure the SiteMinder Login Module in the WebSphere Application Server using the WebSphere Administrative Console. General information about configuring Login Modules is available in the WebSphere documentation.

To configure the WebSphere Application Server to use the SiteMinder Login Module:

1. If necessary, start the WebSphere Server and the WebSphere Administrative Console.
2. Select Security, Global Security.
3. Under authentication, select JAAS Configuration, System Logins.
4. To configure WebSphere to use the SiteMinder Login Module to authenticate system login (RunAs) requests, add the SiteMinder Login Module as a DEFAULT Login Module (see page 78).
5. To configure WebSphere to use the SiteMinder Login Module to authenticate Java client requests, add the SiteMinder Login Module as an RMI_INBOUND Login Module (see page 79).
6. If you are running a WebSphere version earlier than v6.0.2.7, to configure WebSphere to configure the SiteMinder Login Module to assert WebSphere propagation tokens associated with HTTP requests in Subject recreation situations, add the SiteMinder Login Module as an WEB_INBOUND Login Module (see page 80).
7. Click Apply to save your changes.To save changes, click System Administration and Save Changes to the Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

SiteMinder Login Module (see page 15)
Configuring the Login Module, SiteMinder-Side (see page 64)
Adding the SiteMinder Login Module as a WebSphere DEFAULT Login Module (see page 78)
Adding the SiteMinder Login Module as a WebSphere WEB_INBOUND Login Module (see page 79)
Adding the SiteMinder Login Module as a WebSphere RMI_INBOUND Login Module (see page 80)

Adding the SiteMinder Login Module as a WebSphere DEFAULT Login Module

To configure WebSphere to use the SiteMinder Login Module as a DEFAULT JAAS Login Module to handle System Login (J2EE RunAs) requests:

1. Navigate to the Global security>System login configuration page in the WebSphere Administrative Console.
2. Select DEFAULT from the list of JAAS login configurations.
3. On the DEFAULT page, under Additional Properties, select JAAS login modules.
4. Under General Properties on the New page, enter the SiteMinder Login Module Module class name:

   com.netegrity.siteminder.websphere.auth.SmLoginModule
5. Ensure that **REQUIRED** is selected from the Authentication strategy drop-down list.
6. Click Apply.
7. Under Additional Properties, select Custom properties.
8. On the JAAS Login Modules page, click New.
9. Under General Properties on the New page, enter the following:
   - Name: **loginModuleRealmKey**
   - Value: **SystemAuthResource** (name of the Agent configuration parameter whose value specifies for the resource filter in the realm that you created for System Login requests).
   - Description: Optionally, a description.
10. Click Apply to save your changes.
11. Back on the JAAS Login Modules page, click Set Order.
12. Under General Properties on the JAAS Login Module Order page, move the SiteMinder Login Module to be the first Login Module:
    a. Select the com.netegrity.siteminder.websphere.auth.SmLoginModule entry
    b. Move it to the top of the order list.
    c. Click Apply to save your changes.
13. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

    Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

Note: When you configure the SiteMinder Login Module as a WebSphere DEFAULT Login Module, you must ensure that the WebSphere Global Security (see page 73) Cache timeout is set to a value less than 3 times the SiteMinder Agent key rollover period defined on the Policy Server. Thus, if Agent key rollover period is one hour, the Cache timeout value should be less than 180 minutes.

More information:

Configuring the Login Module to Handle System Login Requests (see page 66)
Setting the SystemAuthResource Agent Configuration Parameter (see page 66)
Enabling WebSphere Global Security (see page 73)

Adding the SiteMinder Login Module as a WebSphere WEB_INBOUND Login Module

Note: If you are running IBM WebSphere Application Server v6.0.2.7 (or later) you can ignore this section – the subject propagation issue that made it necessary is no longer applicable in your deployment.

To configure WebSphere to use the SiteMinder Login Module as a WEB_INBOUND JAAS Login Module to assert WebSphere propagation tokens associated with HTTP requests in Subject recreation (see page 16) situations:

1. Navigate to the Global security>System login configuration page in the WebSphere Administrative Console.
2. Select WEB_INBOUND from the list of JAAS login configurations.
3. On the WEB_INBOUND page, under Additional Properties, select JAAS login modules.
4. Under General Properties on the New page, enter the SiteMinder Login Module Module class name:

   com.netegrity.siteminder.websphere.auth.SmLoginModule

5. Ensure that **REQUIRED** is selected from the Authentication strategy drop-down list.
6. Click Apply.
7. Under Additional Properties, select Custom properties.
8. On the JAAS Login Modules page, click New.

9. Under General Properties on the New page, enter the following:
   - Name: **loginModuleRealmKey**
   - Value: **SystemAuthResource** (name of the Agent configuration parameter whose value specifies for the resource filter in the realm that you created for System Login requests).
   - Description: Optionally, a description.
10. Under General Properties on the JAAS Login Module Order page, move the SiteMinder Login Module to be the last Login Module:
    a. Select the com.netegrity.siteminder.websphere.auth.SmLoginModule entry
    b. Move it to the bottom of the order list.
    c. Click Apply to save your changes.
11. Click Apply to save your changes.
12. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

    Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

Recreating Subjects by Asserting WebSphere Propagation Tokens (see page 16)

Adding the SiteMinder Login Module as a WebSphere RMI_INBOUND Login Module

To configure WebSphere to use the SiteMinder Login Module as an RMI_INBOUND JAAS Login Module to handle Java client requests:

1. Navigate to the Global security>System login configuration page in the WebSphere Administrative Console.
2. Select RMI_INBOUND from the list of JAAS login configurations.
3. On the RMI_INBOUND page, under Additional Properties, select JAAS login modules.
4. Under General Properties on the New page, enter the SiteMinder Login Module Module class name:

   com.netegrity.siteminder.websphere.auth.SmLoginModule
5. Ensure that REQUIRED is selected from the Authentication strategy drop-down list.

6. Click Apply.
7. Under Additional Properties, select Custom properties.
8. On the JAAS Login Modules page, click New.
9. Under General Properties on the New page, enter the following:
   - Name: **loginModuleRealmKey**
   - Value: **RmiAuthResource** (name of the Agent configuration parameter whose value specifies the resource filter in the realm that you created for Java client requests).
   - Description: Optionally, a description.
10. Click Apply to save your changes.
11. Back on the RMI_INBOUND>JAAS Login Modules page, click Set Order.
12. Under General Properties on the JAAS Login Module Order page, if necessary, move the SiteMinder Login Module to be the first Login Module:
    a. Select the com.netegrity.siteminder.websphere.auth.SmLoginModule entry
    b. Move it to the top of the order list.
    c. Click Apply to save your changes.
13. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

    Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

Configuring the Login Module to Handle Java Client Requests (see page 64)
Setting the RmiAuthResource Agent Configuration Parameter (see page 64)

Configuring the SiteMinder JACC Provider in WebSphere

You configure the SiteMinder JACC Provider in the WebSphere Application Server using the WebSphere Administrative Console. General information about configuring JACC Providers is available in the WebSphere documentation.

1. Start the WebSphere Server and the WebSphere Administrative Console (if they are not running already).
2. Select Security, Global Security.
3. Under Authorization, select Authorization providers.

4. Under General properties on the Authorization providers page, select the External authorization using a JACC provider option.
5. Under Related items, select External JACC provider.
6. Under General Properties on the New page, enter the following:
   - Name: SiteMinder JACC Provider
   - Description: Optionally, a description
   - Policy class name: **com.netegrity.siteminder.jacc.policy.SmJaccPolicyProvider14**
   - Policy configuration factory class name: **com.netegrity.siteminder.jacc.policy.SmJaccConfigurationFactory**
   - Deselect the Requires the EJB arguments policy context handler for access decisions option.

   (Leave other fields blank.)
7. To save changes to the master repository, click System Administration and Save Changes to Master Repository.

   Note: Until you save changes to the master repository, the Administrative Console uses a local workspace to track your changes.

More information:

SiteMinder Java Authorization Contract for Containers (JACC) Provider (see page 17)
Configuring the SiteMinder JACC Provider, SiteMinder-Side (see page 69)

What to Do After Completing WebSphere-Side Configuration

After completing WebSphere-side configuration of the SiteMinder Agent modules:

1. Log out of the Administration Console.
2. From a command line or shell in the *WS_HOME*/bin directory, stop and then restart the WebSphere Server.

   To stop the server, you will require the server user ID and Server user password you entered when configuring LDAP as a WebSphere user registry (see page 72). The command is:

   stopServer server1 -username *serveruserID* -password *serveruserpassword*

   To start the server, you do not need a password:

   startServer server1

3. To make sure everything is working as expected, view the log files of the SiteMinder Agent modules, Web Agent, and WebSphere (SystemOut.log, SystemErr.log). In the SiteMinder Agent and WebSphere log files, look for application server errors and errors that begin "SMINFO" to find problems related to the SiteMinder Agent.

   WebSphere's SystemOut.log and SystemErr.log file resides in:

   WS_HOME/profiles/*profile_name*/logs/*server_name*

   The logs indicate should indicate that everything is working correctly. If the logs indicate problems, you should troubleshoot your configuration (see page 123).

4. Verify that your SiteMinder Agent is working correctly (see page 85).
5. Configure SiteMinder authorization policies, if necessary (see page 91).
6. Troubleshoot the configuration (see page 121), if necessary.

Chapter 6: Verifying SiteMinder Agent Installation and Configuration

Use the procedures in this chapter to verify that the a TAI-only SiteMinder Agent for IBM WebSphere deployment is installed and configured correctly in the WebSphere Administrative Console and the Policy Server User Interface.

The test scenarios (one for the TAI-only use case and one for an all-modules solution) outlined use the Snoop servlet example Web application, which is installed by default with WebSphere and accessed using:

http://*fully_qualified_domain_name*:9080/snoop

where *fully_qualified_domain_name* is the name of the machine on which WebSphere is installed. For example:

server1.ca.com

When you access this URL in a Web browser, WebSphere prompts you for credentials using a default realm.

This section contains the following topics:

Setting Up the Snoop Servlet Example (TAI-Only)

Note: Ignore this section if you are configuring an All-modules environment and proceed directly to Accessing the Snoop Servlet in a Web Browser (see page 88).

In this example, the goal is to create a SiteMinder realm using an HTML forms authentication scheme so that the SiteMinder TAI intercepts the HTTP request for the Snoop servlet and challenges the user for credentials and authenticates the user. The SiteMinder TAI's role is to verify that the user is authenticated or has a valid SiteMinder token (SiteMinder session cookie). If the TAI authenticates the user, then WebSphere will also do so because the Policy Server and WebSphere share the same user store. Once this criteria is met, WebSphere authorizes the user to access the Snoop servlet.

To set up the example and protect the Snoop Servlet:

1. Start the Policy Server User Interface.
2. On the System tab, right-click User Directories and select Create User Directory to make a user directory configured to the same LDAP user store as the one used by WebSphere. For more information on creating user directories, see *CA eTrust SiteMinder Policy Design*.
3. Right-click Domains and select Create Domain to create a policy domain that you want to protect. Assign the user directory from Step 2 to this domain. For more information on creating domains, see *CA eTrust SiteMinder Policy Design*.
4. On the Domains tab, right-click the domain from Step 3 and then select Create Realm.
5. On the SiteMinder Realm dialog, enter the following information:
   - Name: Default Snoop Realm
   - Description: Default Snoop Realm
   - Agent: Agent Identity for the SiteMinder TAI. This is the Agent name value specified for the DefaultAgentName parameter in the Agent Configuration Object used for the SiteMinder TAI.
   - Resource Filter: /snoop
   - Authentication Scheme: Forms Authentication Scheme

     Forms authentication must be hosted on the Web Agent. For instructions on creating an HTML Forms authentication scheme, see *CA eTrust SiteMinder Policy Design*.
   - Default Resource Protection: Protected
6. Click Apply and OK.

Setting Up the Snoop Servlet Example (All Modules)

Note: Ignore this section if you are configuring a TAI-only environment and proceed directly to Accessing the Snoop Servlet in a Web Browser (see page 88).

In this example, the goal is to create a SiteMinder realm using an HTML forms authentication scheme so that the SiteMinder TAI intercepts the HTTP request for the Snoop servlet and challenges the user for credentials and authenticates the user. The SiteMinder TAI's role is to verify that the user is authenticated or has a valid SiteMinder token (SiteMinder session cookie). If the TAI authenticates the user, then WebSphere will also do so because the Policy Server and WebSphere share the same user store.

Once this criteria is met, the configured *SiteMinder JACC Provider* authorizes the user to access the Snoop servlet.

To set up the example and protect the Snoop Servlet:

1. Start the Policy Server User Interface.
2. On the System tab, right-click User Directories and select Create User Directory to make a user directory configured to the same LDAP user store as the one used by WebSphere. For more information on creating user directories, see *CA eTrust SiteMinder Policy Design*.
3. Right-click Domains and select Create Domain to create a policy domain that you want to protect. Assign the user directory from Step 2 to this domain. For more information on creating domains, see *CA eTrust SiteMinder Policy Design*.
4. On the Domains tab, right-click the domain from Step 3 and then select Create Realm.
5. On the SiteMinder Realm dialog, enter the following information, then click OK:
   - Name: Default Snoop Realm
   - Description: Default Snoop Realm
   - Agent: Agent identity for the SiteMinder Agent or, if using one Agent Configuration Object/Agent identity for each SiteMinder Agent module, the name of the Agent group that contains them.
   - Resource Filter: /snoop
   - Authentication Scheme: Forms Authentication Scheme

     Forms authentication must be hosted on the Web Agent. For instructions on creating an HTML Forms authentication scheme, see *CA eTrust SiteMinder Policy Design*.
   - Default Resource Protection: Protected

6. Right click the Default Snoop Realm entry and select Create Rule under Realm.
7. In the Rule Properties dialog, enter the following information for the rule, then click OK.
   - Name: Snoop Protection Rule
   - Resource: *
   - Action: Make sure that the Web Agent Actions radio button is selected, and the GET action is highlighted.
8. Under the domain in the left navigation pane, right-click Policies and select Create Policy.
9. In the Policy Properties dialog:
   a. In the Name field, enter: Snoop Access Policy
   b. In the Users tab, add users or groups of users that are allowed access to the Snoop servlet.

      For information on adding users to a policy, see the Policies chapter in *CA eTrust SiteMinder Policy Design*.
   c. In the Rules tab, select the Snoop Protection Rule that you created in steps 6 and 7.
   d. Click OK to save the Policy.

Accessing the Snoop Servlet in a Web Browser

After setting up the Snoop servlet example for your SiteMinder Agent configuration in the SiteMinder Policy Server, access the Snoop servlet.

To access the Snoop servlet in a Web browser:

1. Make sure the Policy Server, Web server, and WebSphere are running.
2. Make sure the Web Agent and SiteMinder Agent module(s) are enabled (the EnableWebAgent parameter is set to "YES" in the WebAgent.conf associated with the Web Agent and the Agent configuration files associated with the SiteMinder Agent module or modules).

   If they are not enabled, set the parameter to YES, and then restart the Web server and Web Agent.
3. In a browser, access the Snoop servlet at the following URL:
   http://*fully_qualified_domain_name*:*port*/snoop

   where *fully_qualified_domain_name* is the name of the machine where WebSphere is installed and *port* is its port number. For example:

   server2.ca.com:9080

Using the HTML Forms authentication scheme, the Web Agent should challenge you for credentials through the Default Snoop Realm. Once you are authorized by WebSphere or the SiteMinder JACC Provider, you are granted access to the Snoop servlet on the WebSphere server.

Note: In the all-modules scenario, you will be redirected over HTTPS and receive a security warning because the transport is also protected in that configuration. Click Yes to proceed to the Snoop servlet.

To make sure everything is working as expected, view the log files of the SiteMinder Agent modules and Web Agent. The logs should indicate that everything is working correctly. If the logs indicate problems, you should troubleshoot your configuration (see page 123).

Chapter 7: Configuring Policies for the SiteMinder Agent

This chapter describes how to configure SiteMinder policies to support the SiteMinder Agent for IBM WebSphere.

This section contains the following topics:

Configuring SiteMinder Policies to Support J2EE Roles

You can configure the SiteMinder JACC Provider to support WebSphere J2EE roles by mapping those roles to users and groups defined in a SiteMinder user directory.

Note: The SiteMinder Agent only supports roles that have global scope across all applications. Application-scoped roles (that is, where the same role name is bound to two different sets of users or groups for two different applications for use by programmatic calls) are not supported. Note also that roles defined in web.xml are also not supported.

To configure the SiteMinder JACC Provider to handle J2EE authorization roles, configure:

- A realm, named **SmJaccRoles**, that holds rules that map to roles (see page 92)
- Within the SmJaccRoles realm, a role-mapping rule that corresponds to each role you wish to support (see page 92)
- For each configured role-mapping rule, a policy that binds users to the mapped role (see page 92)

Configuring the SmJaccRoles Realm

To support J2EE role-mapping for SiteMinder authorization policies, configure an SmJaccRoles realm with the following properties:

Property	Value
Name	SmJaccRoles
Resource Filter	/SmJaccRoles
Agent	The Agent Identity associated with the SiteMinder JACC Provider
Authentication Scheme	Basic
Default Resource Protection	Protected

Configuring Role-Mapping Rules

For each role that you need to support when configuring SiteMinder authorization policies, configure a role-mapping rule as described in the following table:

Property	Description	Example Value
Name	Name of the role that the rule represents	managers
Agent	None (do not specify)	N/A
Resource Filter	Name of the role that the rule represents	/managers
Rule Action	Get	Get

Configuring Role-Mapping Policies

To finish configuring SiteMinder to support J2EE roles for authorization, configure role-mapping policies. For each J2EE role, configure a rule that includes:

- The users and/or groups from the configured SiteMinder user directory that you want to associate with the role.
- The corresponding role-mapping rule.

Resource Mapping

The Resource field in a SiteMinder rule specifies the resource that is the subject of the rule. The complete resource specification (shown by the Effective Resource field on the Rule dialog) is a concatenation of the values of the Resource Filter of the parent realm (or realms in a nested realm environment) and the Resource field of the rule itself. Resources must be defined using special mapping conventions.

This section describes the SiteMinder resource mapping for WebSphere resources. This mapping provides a means of representing WebSphere resources in the realms and rules that make up your authorization policies.

Web Application Resources

To protect a WebSphere Web container (URI-based) resource, the SiteMinder resource must specify the following parameters (in the order shown):

/contextPath/[resourceName]

where:

Parameter Name	Description	Field Value Example
contextPath	Context-path of the Web application servicing this URI.	/sm/mywebapp
resourcePath (Optional)	The relative path to the resource requested. Multiple path elements should be treated as separate slash(/)-delimited parameters.	/foo/bar/my.jsp?a=b

For example, for a server application with the following properties:

contextPath=/sm/mywebapp, resourcePath=/foo/bar/my.jsp?a=b

The complete resource mapping (effective resource) would be:

/sm/mywebapp/foo/bar/my.jsp

Note: If you configure the top-level resource as protected (omitting the uri parameter when configuring the resource filter), WebSphere assumes that you also wish to protect the transport (see page 94) for that Web application. The application and all its resources are therefore only available over HTTPS.

More information:

Creating Realms for Challenged Requests (see page 61)

Configuring HTTP Transport Guarantees for Web Application Resources

In accordance with the JSR-115 specification JACC Policy Decision and Enforcement Subcontract, you can configure the SiteMinder JACC Provider to secure transport guarantees for any HTTP accessible resource using J2EE user data permissions.

For example, if the Servlet /Snoop is only to be made available for access over HTTPS for actions GET and POST, the security configuration for "/Snoop" should consist of a J2EE user data constraint with value CONFIDENTIAL for those actions.

To configure an HTTP transport guarantee for an HTTP resource, append its SiteMinder resource specification with the term /CONFIDENTIAL:

/*contextPath*/*[resourcePath]*/CONFIDENTIAL

where:

Parameter Name	Description	Field Value Example
contextPath	Context-path of the application servicing the URI.	/sm/mywebapp/

Parameter Name	Description	Field Value Example
resourcePath (optional)	The relative path to the resource requested. Multiple path elements should be treated as separate slash(/)-delimited parameters.	/foo/bar/my.jsp?a=b

Note: If you omit the resourcePath parameter and specify only the contextPath, all resources associated with the specified Web application are subject to the transport guarantee and are therefore only accessible over HTTPS.

For example,

/sm/mywebapp/CONFIDENTIAL

/sm/mywebapp/foo/bar/my.jsp/CONFIDENTIAL

Mapping EJB Resources

To protect a WebSphere EJB resource, resource_type_filter must specify the following parameters (in the order shown):

/ejb/methodInterface/method/methodParams

where:

Parameter Name	Description	Field Value Example
ejb	Name of the EJB	MyEJB
methodInterface	Method interface invoked on the EJB	Home
method	Method executed on the EJB	myMethod
methodParams	Arguments in the signature of the EJB method. Multiple arguments should be treated as separate comma-delimited parameters.	java.lang.String, int

For example, for an EJB application with the following properties:

ejb=myEJB, methodInterface=Home, method=myMethod, methodParams=(java.lang.String, int)

The complete resource mapping (effective resource) would be:

/MyEJB/Home/myMethod/java.lang.String, int

More information:

Configuring the Login Module, SiteMinder-Side (see page 64)

Configuring Rules for the JACC Provider

A rule identifies specific resources within a realm and whether to allow or deny access to those resources. Rules are the parts of policies (see page 99) that determine precisely which resources are protected, and which types of actions should cause the rule to fire.

A rule is required to allow resource requests to be passed to protected WebSphere resources.

You configure one or more rules for the SiteMinder JACC Provider that identify:

- A specific resource to protect. (Using SiteMinder resource mapping for WebSphere resources to map a WebSphere resource to a SiteMinder representation.)
- The Agent action that will cause the rule to fire (Any appropriate action, such as Post or Get for URL resources; the Get action for all other resource types).
- Whether to allow or deny access to the specified resource when the rule is fired.

For example, a rule can specify that all EJB resources in a realm are protected for Get Agent actions. When a client attempts to access these resources, the rule fires and the policy containing the rule determines whether the consumer application can access the protected EJB application.

For more information about creating rules, see *CA eTrust SiteMinder Policy Design*.

Configuring Authentication and Authorization Responses

The SiteMinder Agent makes responses available for use in J2EE components. Responses pass user attributes, DN attributes, static text, or customized active responses from the Policy Server to the SiteMinder Agent. The Policy Server returns two responses:

- **Authentication Responses**

  During authentication, these Policy Server responses are returned to the SiteMinder Agent, which then attaches them to the SiteMinder Principal for use by resources in both containers such as Servlets, JSPs in their corresponding J2EE applications, and by EJB container resources.

- **Authorization Responses**

  During authorization, these Policy Server responses are returned to the SiteMinder JACC Provider, which places them in an HTTP request attribute for use with HTTP requests *only*; they are not attached to the SiteMinder Principal. Authorization responses are not therefore available for use with EJB container requests.

Configuring SiteMinder Policies to Support User Mapping (Optional)

To support an environment in which SiteMinder is responsible for user authentication but SiteMinder and WebSphere are not configured to authenticate/authorize users against the same user store, you must create user mapping policies (see page 18) consisting of the following policy objects:

- In the first configured policy realm (for Web or EJB resources) that a user accesses when they cross over to WebSphere:
  - An OnAuthAccept rule that will fire whenever a user is successfully authenticated
  - An authentication response that returns the mapped identity that the SiteMinder Agent will propagate to WebSphere
- For each user mapping rule/response pair, a policy that binds that pair and users.

Note: You can also use global rules and responses.

To create a user mapping policy:

1. Open the SiteMinder Policy Server User Interface.
2. Click the Domains tab and open the Domain you created for the SiteMinder Agent for IBM WebSphere.
3. For each configured SiteMinder TAI and SiteMinder Login Module policy realm, configure an OnAuthAccept authentication rule:
   a. Right-click the name of the appropriate TAI or Login Module policy realm and select Create Rule.
   b. On the Rule dialog, select the Authentication events option under Actions and then choose the OnAuthAccept event from the drop-down list.
   c. Click OK.
4. Configure a user mapping response:
   a. Right-click Responses and select Create Response.
   b. On the Response dialog, enter a Name and Description for the response and click Create (response attribute).
   c. On the Attribute Setup tab of the Response Attribute Editor dialog, enter the following information:
      – Attribute: Select the HTTP Header Variable
      – Variable Name: _SM_MAPPED_USER
      – Variable Value: Any text that is a static attribute, DN attribute, or an active response that resolves to a user present in the WebSphere user store.

      Note: If you are upgrading from an earlier SiteMinder TAI implementation, you should change the Variable Name used in your user mapping response from_SM_WAS_ID to _SM_MAPPED_USER. The _SM_WAS_ID variable is deprecated at this release.

   Click OK and click OK again to close the Response dialog.
5. Configure an authentication policy:
   a. Right-click Policies and select Create Policy.
   b. Add all configured user mapping (OnAuthAccept) rules
   c. Associate the user mapping response with each user mapping rule.
   d. Add users to the policy.
   e. Click OK.

More information:

Identity and User Mapping (see page 18)

Configuring Authorization Policies for the SiteMinder Agent

Policies define how clients interact with your WebSphere resources. They bind rules, users, and responses defined within a policy domain that define what should happen when requests are sent to resources defined in a realm.

You configure policies to protect WebSphere resources using the SiteMinder JACC Provider in the same way as you would policies to protect Web resources. Note however, that the following features are not supported:

- Policy expressions
- Impersonation

For more information about creating policies, see *CA eTrust SiteMinder Policy Design*.

Chapter 8: Obtaining SiteMinder Agent Data Programmatically

This chapter tells you how to obtain authentication responses returned in the SiteMinder Principal and authorization responses returned in HTTP request attributes programmatically.

This section contains the following topics:

Common HashMap Response Structure

Both Authentication responses returned in the SiteMinder Principal and authorization responses returned in an HTTP request attribute are in the form of a Java HashMap data structure.

The keys in the HashMap denote the attribute ID's returned from the SiteMinder Agent API, and the values in the HashMap are a list of all the values binding to that key.

For example, if the Policy Server returns two HTTP header responses HEADER1=VALUE1 and HEADER2=VALUE2 during an authorization request, the HashMap will contain a key (Agent API constant denoting that it is a header response) and a value of List with two elements, that is, HEADER1=VALUE1 and HEADER2=VALUE2.

Obtaining Authentication Responses and Other Data from the SiteMinder Principal

You can access authentication responses and other data from the SiteMinder Principal using the SiteMinder User Principal API. This interface, com.netegrity.siteminder.asaframework.common.SmUserPrincipal, provides the following calls:

- getName ()

  Returns the name of a principal.

- getNameDN ()

  Returns the user DN of a principal.

- getSessionID ()

  Returns the session ID of a principal.

- getSessionSpec ()

  Returns the session spec of a principal.

- getAuthDirectoryOid ()

  Returns the Object ID of the user directory a principal was authenticated against.

- getAuthResponses ()

  Returns the responses returned by the Policy Server during authentication in the form of the HashMap described in Common HashMap Response Structure (see page 101).

Note: Your J2SE security policy must be configured to ensure valid permissions for access to the Subject. For example:

```
grant codebase "file:myapp.war" {
   permission javax.security.auth.AuthPermission "wssecurity.getCallerSubject";
};
```

The following code snippet shows how to obtain the SiteMinder Principal:

```
public void service(HttpServletRequest request, HttpServletResponse response)
     throws ServletException, IOException

{
            …

            javax.security.auth.Subject subject =
            com.ibm.websphere.security.auth.WSSubject.getCallerSubject ();

            java.util.Set principals = subject.getPrincipals
            (com.netegrity.siteminder.asaframework.common.SmUserPrincipal.class);

            java.util.Iterator i = principals.iterator();
            while (i.hasNext())
            {
            SmUserPrincipal smUser = (SmUserPrincipal)i.next();
            // Get Authentication Responses
            HashMap authResponseMap = smUser.getAuthResponses();
            }

            …
}
```

Obtaining Authorization Responses for Web Requests from HTTP Request Attributes

Authorization responses are set in the com.ca.siteminder.asa.SmAzResponses HTTP request attribute in the form of the HashMap described in Common HashMap Response Structure (see page 101).

The following code snippet shows how to obtain the response HashMap from the request object.

```
public void service(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException

{
        …
        Object attribute = request.getAttribute (“com.ca.siteminder.asa.SmAzResponses”);
        if (attribute != null)
        {
                // do some processing

        }
….
}
```

Chapter 9: Session Handling

This chapter describes how to configure session parameters, such as timeout values, to handle the differences between SiteMinder and WebSphere session handling.

This section contains the following topics:

Session Synchronization Between WebSphere and the SiteMinder Agent

The SiteMinder Agent for IBM WebSphere does not support SiteMinder session management. (However, each SiteMinder Agent module honors SiteMinder session idle and max timeouts.)

To interoperate with WebSphere SSO, WebSphere SSO must be enabled. When WebSphere SSO is enabled, the SiteMinder TAI is not invoked for subsequent HTTP requests once the WebSphere SSO token is set in the HTTP client. Therefore, the SiteMinder TAI cannot intercept every HTTP request to enforce SiteMinder session management by updating the SiteMinder session cookie. Based on this, WebSphere is the session controller for any user session within the WebSphere environment; thus, sessions must be synchronized between WebSphere and the SiteMinder Agent.

Handling Timeouts

In the case of idle timeouts, the SiteMinder idle timeouts for every realm must be set greater than or equal to the WebSphere LTPAToken timeout. This is because if the SiteMinder idle timeout is less than the LTPAToken, then users moving from WebSphere to SiteMinder will be: challenged for credentials if they enter WebSphere for the first time after the idle timeout; after that, the TAI is not invoked and the JACC denies all request with a 403 error.

In the case where the SiteMinder idle timeout is greater than the LTPAToken timeout, the SiteMinder session ticket will be valid even though the LTPAToken has timed out. This would result in the existing SiteMinder session ticket being propagated back to SiteMinder and eventually this would result in skews between SiteMinder and WebSphere timeouts. In this case, WebSphere will force a rechallenge and the TAI will create a new SiteMinder Principal with refreshed last access times.

In the case of SiteMinder maximum timeouts, the maximum timeout must be a multiple of idle timeout. For example, if *idle_timeout* = *LTPA_cookie_timeout* = 1 hour, then *max_timeout* must be (*n* * *idle_timeout*) where *n* = 1, 2, 3, 4, and so on. This forces WebSphere to trigger the TAI again to challenge the user for updated credentials.

Note: Max timeout settings can result in timeout skew; if the timeouts are not synchronized and a user session hits the maximum timeout, the user must close the browser session and open a new one.

Handling Single Log Off

SiteMinder creates the SiteMinder session cookie for SSO and WebSphere creates the LTPAToken for SSO. In order for a proper logout, users must log off in both WebSphere and SiteMinder therefore both SiteMinder session cookies and the LTPAToken must be either deleted from the browser session or the cookie value should indicate that the user is logged off.

Chapter 10: Logging

This chapter describes how to configure logging the the SiteMinder Agent for IBM WebSphere.

This section contains the following topics:

Log Files

Two log files provide important information during SiteMinder Agent configuration:

- SiteMinder Agent log files—Logs SiteMinder Agent error and processing messages to a file only, not to a console
- Default SiteMinder Agent log file—Logs messages regarding the connection between the SiteMinder Agent and Policy Server

SiteMinder Agent Log File

This logging function enables you to monitor the performance of a particular SiteMinder Agent module instance. You can configure the Agent instance to log messages to a file, but not to a command prompt window.

Set up SiteMinder Agent logging either:

- Locally, in a SiteMinder Agent module's Agent configuration file.
- Centrally, using the Agent Configuration Object in the Policy Server User Interface.

In a complex environment, you could have several SiteMinder Agent instances installed on the same machine for multiple WebSphere instances that are all logging separately and sharing the same connection to the Policy Server while logging to the same default SiteMinder Agent log file.

Note: Settings in the local Agent configuration file take precedence over the log settings in the Agent Configuration Object from the Policy Server User Interface.

More information:

Fine-Tuning the Agent Configuration Setup (see page 52)
Configuring the Class Loader for the SiteMinder Agent Logger (see page 74)

Default SiteMinder Agent Log File

This logging function enables you to monitor the connection between the SiteMinder Agent and the Policy Server. The file logs SiteMinder Agent startup messages and shows whether the Agent made a successful connection with the Policy Server. It also logs messages associated with dynamic agent configuration.

Set up default SiteMinder Agent logging in the smagent.properties file (see page 50); it cannot be configured by using the Agent Configuration Object in the Policy Server User Interface.

More information:

Editing smagent.properties (see page 51)

Recording Messages to the Default SiteMinder Agent Log File

To record messages to the default SiteMinder Agent log file, edit the smagent.properties file:

1. Set the logfile parameter to YES:

   logfile="YES"

2. Specify a path and file name for the logfilename parameter.

   For example:

   logfilename="/opt/WebSphere/AppServer/smwasasa/logs/*log_file_name*.log"

   where *log_file_name* is the name of your log file.
   For example: SiteminderAgent.log

   The default file name is:
   ASA_HOME/log/SMWASASADefaultLog.log"

Appending Messages to an Existing Log File

To add logging information to an existing default log file instead of rewriting the entire file each time logging is invoked, enable the logappend parameter. For example:

logappend="YES"

Setting the Log Level

The SiteMinder Agent does not support displaying log messages in a console.

Dynamically Updating the SiteMinder Agent Log Files

To dynamically update the SiteMinder Agent log file for a module, you must make the appropriate changes in that module's Agent Configuration Object, found in the Policy Server User Interface. Changes made to the Agent Configuration Object are reflected in the SiteMinder Agent module after the Agent receives the changes at the next polling interval.

For example, to change the log level from 1 to 3:

1. In the Policy Server User Interface, double-click the Agent module's Agent Configuration Object on the Domains tab.
2. Select Log Level and click Edit.
3. Change the log level value from 1 to 3 and click OK.
4. Click OK. The level 3 logging messages are then displayed without restarting the Policy Server or WebSphere.

Rolling Over the Log File

Rollover determines whether the SiteMinder Agent starts a new log file after a specified period of time or after the log file reaches a specified size. Roll over the SiteMinder Agent log file by using time or size, and making the appropriate changes in the Agent Configuration Object from the Policy Server User Interface.

To enable rollover and specify rollover limits, add the following parameters to the Agent Configuration Object:

logrollover = "YES" or "NO"

logrolloversize = *size_in_KB*

where *size_in_KB* is the number of kilobytes you want the file to be before rollover occurs. Rollover does not take affect unless the parameter LOGROLLOVER is set to a "YES" value.

logrollovertime = *rollover_hours*

where *rollover_hours* variable is the number of hours until rollover occurs. For example, 1 is every hour; 12 is every 12 hours; 168 is every week; and 720 is every month. Rollover does not take affect unless the parameter LOGROLLOVER is set to YES.

Note: Use either logrolloversize or logrollovertime. If you use both, rollover by size takes precedence.

Appendix A: SiteMinder Agent Installation and Configuration Files

This section contains the following topics:

SiteMinder Agent Files

The installation program creates several directories, populates them with files, and copies some of the files to the WebSphere Application Server.

The following table lists the directories and files that the installation program creates and populates in the SiteMinder Agent installation location.

Install Location	Files Installed	Description
ASA_HOME	**Windows**: asa-was-uninstall.cmd **UNIX**: asa-was-uninstall.sh	Script to launch the application uninstaller.
ASA_HOME/bin	**Windows**: smreghost.bat **UNIX**: smreghost.sh	SiteMinder tool to register a trusted host

Install Location	Files Installed	Description
ASA_HOME/conf	AsaAgent-assertion.conf AsaAgent-auth.conf AsaAgent-az.conf SmHost.conf smagent.properties	SiteMinder configuration and properties files
ASA_HOME/log	None (empty directory created)	Directory to hold log files for the SiteMinder Agent installation
ASA_HOME/asa-was-uninstall	**Windows**: uninstall.exe **UNIX**: uninstall	Uninstalls the ASA

In addition, the SiteMinder Agent installation program installation program copies the following files to the WebSphere Application Server installation.

Install Location	Files Installed	Description
WS_HOME\lib\ext	smagent.jar smwebsphereasa.jar smlogger.jar smclientclasses.jar log4j.jar	Containers for SiteMinder Agent class files
WS_HOME\java\jre\lib\ext	sm_jsafe.jar sm_jsafeJCE.jar	Container for SiteMinder Agent Java Cryptography Extension (JCE) class files

Modifying Configuration Files

To customize the SiteMinder Agent configuration, you can modify:

- Agent configuration settings (see page 114)
- Trusted Host configuration settings (see page 120)

Guidelines for Modifying Configuration Files

- Do not add extra spaces between these elements of the parameter settings: parameter name, the equals sign (=), and the attribute value.
- Always enter quotation marks around the parameter value.
- Restart the WebSphere Application Server after you have updated and saved configuration files, such as AsaAgent-*module*.conf and SmHost.conf.

  You do not have to restart the WebSphere Application Server after you have updated and saved configuration objects, such as the Agent Configuration Object or the Host Configuration Object.

Agent Configuration Parameters

Agent configuration settings are defined in two locations:

- **Agent Configuration Object—**SiteMinder policy object that holds Agent parameters for an Agent when using central agent configuration. You can create a separate Agent Configuration Object for each Agent module if you want to centrally define different parameters for each module.

  Note: Parameters held in an Agent Configuration Object are dynamic; if you change these settings while the WebSphere server is running, the SiteMinder Agent will pick up the change.

- **Agent Configuration File—**Text file that holds parameters for an Agent. By default, an Agent configuration file is created for each module (AsaAgent-*module*.conf, where *module* is assertion for the TAI, auth for the Login Module, and az for the JACC Provider). However, you can create a single AsaAgent.conf file to provide common parameters for all three modules.

Unless otherwise noted, parameters can be defined in either the Agent Configuration Object or the Agent configuration file depending upon how you have decided to configure your Agent.Fine-Tuning Your Agent Configuration Setup (see page 52) .

Parameter Name	Value	Description
AcceptTPCookie (TAI and Login Module)	YES or NO	(Optional) If set to yes, configures the SiteMinder TAI/SiteMinder Login Module to assert identities from third-party SiteMinder session cookies. Default is No. **Note:** The value you specify for this parameter must match for the SiteMinder TAI, SiteMinder Login Module, and Web Agent.

Parameter Name	Value	Description
AgentConfigObject (Applies only in Agent configuration file)	String	The name of the Agent module's Agent Configuration Object.
AllowLocalConfig (Applies only in the Agent Configuration Object)	YES or NO	If set to yes, parameters set locally in the Agent configuration file take precedence over parameters in the Agent Configuration Object. Default is NO.
AssertionAuthResource (TAI only)	String	(Optional) If you are configuring the TAI to not challenge requests for credentials, this value must match the value specified for the resource filter in the realm that you create for non-challenged requests. Creating Assertion Realm for Non-Challenged Requests. For example: assertionauthresource=/siteminderta i If configuring the TAI to challenge requests for credentials, this value should be NO. Default is NO.
AuthCacheSize (TAI and Login Module)	Number	(Optional) Size of the authentication cache for the SiteMinder TAI or Login Module (in number of entries). For example: authcachesize="1000" Default is 0. To flush this cache, use the Policy Server User Interface.
AzCacheSize (JACC Provider)		(Optional) Size of the authorization cache (in number of entries) for the JACC Provider. For example: authcachesize="1000" Default is 0. To flush this cache, use the Policy Server User Interface.

Parameter Name	Value	Description
CacheTimeout	Number	(Optional) Number of seconds before cache times out. For example: cachetimeout="1000" Default is 600 (10 minutes).
ChallengeForCredentials (TAI)	YES or NO	(Optional) Specifies whether the SiteMinder TAI should challenge for credentials. Default is NO.
CookieDomain (TAI)	String	(Optional) Name of the cookie domain. For example: cookiedomain="ca.com" No default value. For more information, see the cookiedomainscope parameter.
CookieDomainScope (TAI)	Number	(Optional) Further defines the cookie domain for assertion of SiteMinder session cookies by the SiteMinder TAI. The scope determines the number of sections, separated by periods, that make up the domain name. A domain always begins with a period (.) character. For example: cookiedomainscope="2" Default is 0, which takes the domain name specified in the cookiedomain parameter, described in Step 3.
DefaultAgentName (Applies only in the Agent Configuration Object)	String	The Agent identity that the SiteMinder Agent module for which it is set uses when it detects an IP address in a request that does not have an Agent identity assigned to it. By default, the default Agent name is the name of the installed Agent (module).
EnableWebAgent (Applies only in Agent configuration file)	YES or NO	Enables or disables the SiteMinder Agent for WebSphere module for which it is set.

Parameter Name	Value	Description
EncryptAgentName (TAI)	YES or NO	Specifies whether the agent name should be encrypted when redirecting to the SiteMinder Web Agent for SiteMinder TAI credential collection. Default is NO. **Note:** The value you specify for this parameter must match for the SiteMinder TAI and Web Agent.
FccCompatMode (TAI)	YES or NO	(Required for TAI; otherwise optional) Specifies whether to handle backward compatibility of forms credential collection, which the SiteMinder TAI does not support. You must therefore set this parameter to NO for both the SiteMinder TAI and the Web Agent: fcccompatmode="NO" **Note:** The value you specify for this parameter must match for the SiteMinder TAI and Web Agent.
IgnoreExt (JACC Provider)	Comma separated string	(Optional) Species common file extensions (.gif, .jpg, .jpeg, .png, and .class) that the JACC Provider can ignore. The JACC Provider passes requests for files with these extensions directly to WebSphere without authorization. Use this parameter to specify extensions of files that do not require as much security as other resources.
IgnoreQueryData (TAI and JACC Provider)	YES or NO	(Optional) Indicates whether the SiteMinder TAI/JACC Provider should ignore HTTP query data when checking for resource protection. Default is NO. **Note:** The value you specify for this parameter must match for the SiteMinder TAI, JACC Provider, and Web Agent (as applicable).
IpCheck (TAI and JACC Provider)	YES or NO	(Optional) Enables or disables checking of IP addresses by SiteMinder TAI/JACC Provider. Default is YES. **Note:** The value you specify for this parameter must match for the SiteMinder TAI, JACC Provider, and Web Agent (as applicable).

Parameter Name	Value	Description
LogAppend	YES or NO	(Optional) If an existing file is present in the location specified in logfilename, the logappend parameter determines whether to append messages to that file or to overwrite the file. YES appends messages; NO overwrites the file. Default is NO.
LogConsole	YES or NO	(Optional) YES or NO, to enable logging to the console. Default is NO.
LogFile	YES or NO	(Optional) YES or NO, to enable or disable logging to a log file. Default is NO.
LogFileName	String	(Optional) Agent log file path. For example: /opt/WebSphere/AppServer/smwasasa/logs/asa.log
LogLevel	Number	(Optional) 0, 1, 2, or 3, 4, or 5 levels at which log messages are written. Default is 0.
LogRollover	YES or NO	(Optional) If yes, enables logging rollover. Default is NO.
LogRolloverSize	Number	(Optional) Number, in kilobytes (KB), that specifies the size limit of the log file before you want it to rollover. Specify this only if logrollover is set to YES. Positive integer only. The default is 10240 KB (10 MB).
LogRolloverTime	Number	(Optional) Number, in hours, that specifies the duration before you want the log file to rollover. Specify this only if logrollover is set to YES. Positive integer only. The default is 12 hours.
ResourceCacheSize	Number	(Optional) Size (in number of entries) of the cache for resource protection decisions. For example: resourcecachesize="1000" Default is 0. To flush this cache, use the Policy Server User Interface.

Parameter Name	Value	Description
RmiAuthResource (Login Module)	String	(Optional) Specifies the value of the resource filter defined in realm that you create for Java Client requests or "no" if you do not want the Login Module to accept Java client requests. For example: RmiAuthResource=siteminderrmi Default is NO.
ServerErrorFile (TAI)	String	(Optional) Specifies a page to redirect a request to if a processing error is encountered. This can either be an HTTP or local file system resource. For example: servererrorfile="http://server.ca.com:88/errorpage.html" or servererrorfile="C:\smwasasa\errorpages\errorpage.txt" If this setting is not configured, a default message is output to the response when the TAI encounters an error.
SystemAuthResource (Login Module)	String	(Optional) Specifies the value of the resource filter defined in the realm that you create for System Login requests (see page 66) or "no" if you do not want the Login Module to handle System Login requests. For example: SystemAuthResource=sitemindersystem Default is NO

You can specify logging settings in the SiteMinder Agent Configuration Object in the Policy Server User Interface or in the local SiteMinder Agent's Agent configuration file.

Trusted Host Configuration

The SmHost.conf file results from a successful registration of a unique host name as a trusted host. The SiteMinder Agent installation program automatically launches the smreghost registration tool, which in turn creates the SmHost.conf file and places it in the *ASA_HOME*/conf folder.

Sample SmHost.conf file:

```
# Host Registration File - SmHost.conf
# This file contains bootstrap information required by
# the SiteMinder Agent API to connect to Policy Servers
# at startup. Be sure the IP addresses and ports below
# identify valid listening Policy Servers. Please do not
# hand edit the encrypted SharedSecret entry.
hostname="my-was-host"
sharedsecret="{RC2}PvkvYq/yKgBe/VP9GAD06lOOoA"
hostconfigobject="reptile-host-config"
# Add additional bootstrap policy servers here for fault tolerance.
policyserver="172.26.6.123,44441,44442,44443"
requesttimeout="60"
cryptoprovider="BSAFE"
```

For information about trusted hosts and the parameters in the file, see the *CA eTrust SiteMinder Web Agent Installation Guide*.

To register a trusted host outside the SiteMinder Agent installation process, run smreghost through the command line (see page 41).

Enabling and Disabling SiteMinder Agent Modules

After configuration, each SiteMinder Agent for IBM WebSphere is enabled and ready to communicate with the Policy Server to gather management information. When you disable a SiteMinder Agent module, it no longer performs its functions and these default to another configured implementation of the same module or WebSphere's native security mechanism.

Important: If the SiteMinder JACC Provider is configured, you must not disable it – doing so will prevent the WebSphere Application Server from starting.

You disable and enable SiteMinder Agent modules by changing the value of the EnableWebAgent parameter in the corresponding Agent configuration file:

1. Open the Agent configuration file in the *ASA_HOME*\conf directory.
2. Set the EnableWebAgent parameter as follows:
   - To disable the module, set EnableWebAgent to No as follows:

     EnableWebAgent="No"

   - To enable the module, set EnableWebAgent to No:

     EnableWebAgent="Yes"

Note: The EnableWebAgent parameter applies to all of modules that use the Agent configuration file. For example, if you configured the Agent modules to use a single Agent configuration file, setting EnableWebAgent to yes enables all of Agent modules.

Appendix B: Troubleshooting

This chapter contains guidelines for diagnosing problems and specific advice on how to solve the most common ones.

This section contains the following topics:

General Troubleshooting Guidelines

The following general guidelines should help you find solutions to problems:

- Set the SiteMinder Agent and SiteMinder Default Agent logs to level 5.

  In the smagent.properties file, make sure to specify the path to the default SiteMinder Agent log file. For example:
  logfilename="*ASA_HOME*\log*MyDefaultLog.log*"
- Check the SiteMinder Default Agent logs for general connectivity issues between the SiteMinder Agent and Policy Server.
- Check the SiteMinder Agent module logs for event-specific messages.
- Check the Web Agent log if you are using non-basic authentication.
- Check the WebSphere SystemOut.log file, which resides in:
  WS_HOME/profiles/*profile_name*/logs/*server_name*
- Check the WebSphere SystemErr.log file, which also resides in:
  WS_HOME/profiles/*profile_name*/logs/*server_name*

WebSphere Application Server Does Not Start

WebSphere Application Server fails to start.

Log Message	Possible Cause	Proposed Solution
SystemErr.log: 3/14/06 10:56:00:994 IST] 0000000a SystemErr R SMERROR: Unable to create configuration from the administration manager: Failed to create agent configuration for : C:\smwasasa\conf\AsaAgent-az.conf [3/14/06 10:56:00:994 IST] 0000000a SystemErr R SMFATAL: SiteMinder JACC Policy Provider SmJaccPolicyProvider14 unable to instantiate delegating policy provider: SiteMinder JACC ASA initialization error: Unable to create configuration setup from the policy server	SiteMinder JACC Provider is configured and Policy Server is not started.	Ensure that the Policy Server is running.
	Host Configuration Object or Configuration Objects are not correct.	Ensure that Agent configuration objects being used are mentioned correctly in the Agent configuration files.
	smagent.properties not in classpath.	Make sure that the smagent.properties file exists in *WAS_HOME*/profiles/default/properties.
	Relative path not properly set.	Ensure that value specified in the smasa.home sytem property is correct.

Log Message	Possible Cause	Proposed Solution
[3/14/06 11:03:40:029 IST] 0000000a distSecurityC E SECJ0391E: Error when setting the Policy object to the providers policy implementation {0}. The exception is {1}. [3/14/06 11:03:40:107 IST] 0000000a distSecurityC E SECJ0324E: Error during Java 2 Security and Dynamic Policy initialization. The exception is com.ibm.ws.exception.ConfigurationError: co.netegrity.siteminder.jacc.policy.SmJaccPolicyProvider14 at com.ibm.ws.security.core.distSecurityComponentImpl.initializeJaccProxy(distSecurityComponentImpl.java:1150)	SiteMinder JACC Provider not configured properly in WebSphere console.	Retrace the steps of SiteMinder JACC Provider configuration in WebSphere and ensure everything is configured correctly.
[3/14/06 12:34:16:015 IST] 0000000a SystemErr R com.ibm.ws.exception.ConfigurationError: Error during Java 2 Security and Dynamic Policy initialization	SiteMinder Agent jars not available.	Ensure that SiteMinder Agent jars are available under *WAS_HOME*/lib/ext and are in classpath.

Log Message	Possible Cause	Proposed Solution
[3/14/06 12:17:21:791 IST] 0000000a SystemErr R SMERROR: The SiteMinder Login Module is not initialized - failing method: login [3/14/06 12:17:29:103 IST] 0000000a SystemErr R com.ibm.ws.exception.RuntimeError: com.ibm.ws.exception.RuntimeError: SiteMinder Login Module is not initialized - failing method = login at com.ibm.ws.runtime.WsServerImpl.bootServerContainer(WsServerImpl.java:182) at com.ibm.ws.runtime.WsServerImpl.start(WsServerImpl.java:120) at com.ibm.ws.runtime.WsServerImpl.main(WsServerImpl. Login Module Log: SystemAuthResource [14 Mar 2006 12:17:21,791] [P=833713:O=0:CT] [INFO] Started transaction ID 4adbc0ca-098a4d1a-8847fde1-dd3ddf52-e73a27f2-625 in SiteMinder agent Login Module. [14 Mar 2006 12:17:21,791] [P=833713:O=0:CT] [DEBUG] SiteMinder Login Module initializing with realm key = SystemAuthResource [14 Mar 2006 12:17:21,791] [P=833713:O=0:CT] [ERROR] The SiteMinder Login Module is missing agent configuration realm key SystemAuthResource. [14 Mar 2006 12:17:21,791] [P=833713:O=0:CT] [ERROR] The SiteMinder Login Module is not initialized - failing method login.	SystemAuthResource Agent configuration parameter is not specified for the SiteMinder Login Module.	Set the SystemAuthResource Agent configuration parameter for the SiteMinder Login Module.

Log Message	Possible Cause	Proposed Solution
[3/14/06 11:19:46:801 IST] 0000000a SystemErr R SMFATAL: SiteMinder JACC Policy Provider SmJaccPolicyProvider14 unable to instantiate delegating policy provider: SiteMinder JACC Policy Provider agent is not enabled [3/14/06 11:19:47:020 IST] 0000000a SystemErr R com.ibm.ws.exception.ConfigurationError: Error during Java 2 Security and Dynamic Policy initialization at com.ibm.ws.security.core.distSecurityComponentImpl.initializeJava2Sec(distSecurityComponentImpl.java:1244) at com.ibm.ws.secu	SiteMinder JACC Provider is disabled.	Enable the SiteMinder JACC Provider in its Agent configuration file.

Message While Loading JVM

Installation of Siteminder Agent in GUI mode or console mode does not work; a pop-up message appears.

Possible Cause	Solution
Problem with the java VM or with loading the java VM.	Check for any messages that indicate an error occurred while loading the java VM. Set the PATH to java bin directory that comes with WebSphere. For example: C:\Program Files\IBM\WebSphere\AppServer\java\bin See Setting a PATH Variable to the JVM On UNIX Systems (see page 33). Ensure that the Java Cryptography Extension (JCE) patch for the JVM is installed.

Host Registration Fails During Installation

You are unable to register a trusted host during SiteMinder Agent installation. During installation, make sure the Policy Server is running—the installation program connects to it to create a trusted host.

Possible Cause	Solution
The JVM has not been patched for unlimited cryptography with the with the Java Cryptography Extension (JCE) package.	Check for messages that indicate that the host could not be registered. Patch the JVM for unlimited cryptography with the with the Java Cryptography Extension (JCE) package. For more information, see Required Software Patches (see page 32).
Host configuration object has not been configured or Policy Server is not running.	Check for any messages that indicate the host could not be registered. Add a host configuration object and run smreghost tool separately from the installation procedure. To run this tool, see the *CA eTrust SiteMinder Web Agent Installation Guide*.
Trusted Host Name already exists in the Policy Server.	Check for any messages that indicate the host could not be registered. Configure with another trusted host name or delete the already existing one.

WebSphere Starts With No Indication That SiteMinder Agent Module Loads

When you start WebSphere, you do not see any indication that the SiteMinder Agent is loaded; WebSphere does not seem to recognize the SiteMinder Agent.

Possible Cause	Solution
SiteMinder Agent module is not correctly configured in the WebSphere Administrative Console.	Make sure the failing SiteMinder Agent module is configured and saved correctly in the WebSphere Administrative Console. Complete the configuration procedure for the appropriate module: ■ Configuring the SiteMinder TAI in WebSphere (see page 75) ■ Configuring the Login Module in WebSphere (see page 77) ■ Configuring the SiteMinder JACC Provider in WebSphere (see page 81)

SiteMinder Agent Initialization Fails

Any SiteMinder Agent module might fail to initialize for several reasons.

Possible Cause	Solution
The SiteMinder Agent module cannot establish an agent connection to the Policy Server.	Check the SiteMinder Default Agent log for any message indicating connection problems. Make sure the SiteMinder Agent module's Agent configuration file points to the correct SmHost.conf file. Also, make sure the Agent module connects to the Policy Server. Test the connection by using the Policy Server's SiteMinder Test tool.
SiteMinder Agent classes are not available.	Check the SystemOut.log and SystemErr.log files. Make sure the SiteMinder Agent .jar files exist in the WebSphere library directory *WS_HOME*\lib\ext. For more information, see SiteMinder Agent Files.
The smagent.properties file is not installed on WebSphere or is not in the correct location on WebSphere.	Check the SystemOut.log and SystemErr.log files. Make sure you have copied the smagent.properties file to the correct location on WebSphere. For more information, see Copying the smagent.properties File to WebSphere (see page 50)
Incorrect path specified to the Agent configuration file.	Check the SystemOut.log and SystemErr.log files. Make sure the correct path is specified to the failing module's Agent configuration file in the smagent.properties (see page 50) file.
SiteMinder host registration is configured incorrectly.	Check the SystemOut.log file, SystemErr.log file and the SiteMinder Default Agent log. Verify the host configuration object and the trusted host in the SmHost.conf file also exist in the Policy Server.

Possible Cause	Solution
Agent Configuration Object or the Default Agent Name are configured incorrectly.	Check the SiteMinder Agent log. Make sure the AgentConfigObject parameter listed in the Agent configuration file file exists and is set correctly, and that the DefaultAgentName of the Agent Configuration Object in the Policy Server User Interface is correct.

SiteMinder TAI Forms Authentication Scheme Failures

Forms authentication schemes configured to challenge users for the SiteMinder TAI result in authentication failures. Symptoms of Forms authentication failures might include:

- The forms authentication scheme is not working with the Web Agent you specified for redirecting FCC (forms credential collectors).
- After providing credentials to Forms authentication, WebSphere challenges again.

Log Message	Possible Cause	Proposed Solution
Web Agent log: DoIsProtected - Policy Server authorization logs may contain more detail. loginUser - Exiting with HTTP 500 Server Error: 20-0003.	EncryptAgentName= YES in TAI Agent Configuration object EncryptAgentName = NO in Web Agent Configuration object	Set the same value for the EncryptAgentName Agent configuration parameter for both the SiteMinder TAI and Web Agent. That is, set both to yes or both to no.

Log Message	Possible Cause	Proposed Solution
Web Agent log: SmCredCore::ResolveAgentName - Error decrypting agent name. loginUser - Exiting with HTTP 500 Server Error: 00-0001	EncryptAgentName= NO in TAI Agent Configuration object EncryptAgentName = YES in Web Agent Configuration object	
	The FCCcompatmode parameter is not set correctly.	The FCCcompatmode parameter should always be set to NO for the SiteMinder Agent. For more information about this parameter, see Disabling FCC Compatibility and Legacy Encoding (see page 60).
	The Policy Server domain has multiple user directories with the same user.	Move the user directory name you configured with WebSphere to first place on the list.

Identity Obtained by TAI Not Propagated to WebSphere

Signing in to the Web Agent is successful, but the identity is not propagated to WebSphere (SiteMinder TAI module seems to have initialized successfully, but when you authenticate with the Web Agent, WebSphere does not recognize you).

Possible Cause	Solution
No native security constraint against the WebSphere resource.	The SiteMinder TAI is not triggered unless a security constraint is issued against the URL that is being accessed. Review the Web deployment descriptor (web.xml) to determine the security constraints in the application.
The Assertion realm might not be protected if the challengeforcredentials parameter is set to NO. The resource might not be protected if the challengeforcredentials parameter is set to YES.	Determine the setting for the parameter.

SiteMinder Agent Initializes but WebSphere Challenges Security

The SiteMinder Agent appears to have initialized successfully, but upon authentication with the Web Agent, the user is challenged by WebSphere native security.

Possible Cause	Solution
Non-SSL requests are rejected due to transport requirement.	Check the <transport-guarantee> element in the web.xml deployment descriptor. Communication might require SSL usage.
SiteMinder Agent is not enabled.	Check the SiteMinder Default Agent Log or SystemOut.log. Set EnableWebAgent parameter to "YES" in the asaagent.conf file. See Enabling or Disabling the SiteMinder Agent.
A SiteMinder user is not mapped to a user in the WebSphere active registry.	Check the SystemOut.log and SystemErr.log — although no specific message is displayed, other messages in the SystemOut.log file should give an indication of behavior. Check the user mapping between the two directories and make sure that user exists in both.
The Assertion realm might not be protected if the challengeforcredentials parameter is set to NO. The resource might not be protected if the challengeforcredentials parameter is set to YES.	Determine the setting for the parameter.

User Not Challenged for Credentials

User is granted access to a resource without being challenged or receives an HTTP 403: Forbidden Error without being challenged.

Log Message	Possible Cause	Proposed Solution
SiteMinder JACC Provider logs indicate that resource is not protected.	The resource might not be protected by the SiteMinder JACC Provider if the request contains query data and the IgnoreQueryData Agent configuration parameter is set to NO.	Create a policy protecting the resource and the query data in SiteMinder Policy Server or change the value of the IgnoreQueryData Agent configuration parameter to yes for the SiteMinder JACC Provider.
Check SiteMinder JACC Provider and SiteMinder TAI logs.	SiteMinder JACC Provider ignores the request.	Check if the extension for the requested resource is configured in IgnoreEXt parameter. If it is, remove it.

Log Message	Possible Cause	Proposed Solution
		The final resource being accessed might be accessed using forward or include (that is, server side redirect); the SiteMinder Agent ignores these requests.

Log Message	Possible Cause	Proposed Solution
Check SiteMinder JACC Provider and SiteMinder TAI logs.	IgnoreQueryData Agent configuration parameter is set to YES for SiteMinder JACC Provider but IgnoreQueryData is set to NO for the SiteMinder TAI If a request contains query data and SiteMinder JACC Provider is configured to ignore the query data, it considers the resource protected and the request is redirected to the SiteMinder TAI. The SiteMinder TAI is configured not to ignore the query data and thus considers the resource not protected and does not create SiteMinder Principal object. The JACC then denies the user access to the resource.	Configure matching values for the IgnoreQueryData Agent configuration parameter for the SiteMinder TAI and SiteMinder JACC Provider.

SiteMinder TAI in No Challenge Mode Not Intercepting Requests

SiteMinder TAI configured not to challenge for credentials; SiteMinder TAI log file shows that module is not intercepting requests.

Possible Cause	Proposed Solution
AssertionAuthResource Agent configuration parameter incorrectly set	AssertionAuthResource Agent configuration parameter needs to be set to a value (for example, /assertionrealm); a realm must be configured with the resource filter set to the exact same value (that is, /assertionrealm in this example case).

500 Error Accessing Any Servlet/EJB

Requests for any servlet/EJB resource results in an HTTP 500: Internal server error.

Log Message	Possible Cause	Proposed Solution
Check SystemOut.log or trace.log or SystemErr.log for Java 2 security violation errors, permission problems.	Incorrect permissions in was.policy file.	Ensure that was.policy file for the application being accessed contains the adequate java permissions to allow a user to access the resource.

User Challenged for Credentials Before WebSphere Session Expires

The user is challenged by SiteMinder before the WebSphere session expires. If the WebSphere session times out before SiteMinder, the TAI will revalidate the user (only if WebSphere SSO is off).

Possible Cause	Solution
The SiteMinder session time is shorter than the WebSphere session time.	Check the SystemOut.log file for any indication that the session has expired and user will be challenged. Set the max and idle Timeout so that the SiteMinder session is greater than the WebSphere session. Manage the Web session times through the Web Agent realm. The Session timeout parameter is in Global Security section of the WebSphere Administrative Console. Set this parameter to the same duration as the session max timeout even if the SiteMinder session expires. If the user re-authenticates, the WebSphere session is renewed. Synchronize WebSphere and SiteMinder session times.

User Mapping Not Working for Login Module-Protected Resources

User mapping not working; SiteMinder Login Module and SiteMinder JACC Provider logs show that the user is not being validated properly.

Possible Cause	Proposed Solution
SiteMinder Login Module configured in wrong order in WebSphere.	Ensure the SiteMinder Login Module is configured first in order for all configured profiles.

Resetting the Level of the IIS Web Agent

When you use an IIS Web Server as a proxy for WebSphere, the WebSphere plug-in installation program sets the WebSphere ISAPI filter at a higher priority than the IIS Web Agent, which is incorrect. Therefore, you must set the Web Agent at a higher level than the WebSphere ISAPI filter.

To reset the level of the IIS Web Agent:

1. Start the IIS Microsoft Management Console.
2. Right-click the node and select Properties.
3. Select Edit, while leaving the WWW service in the drop-down menu.
4. Select the ISAPI Filters tab.
5. Highlight the sePlugins filter and move the sePlugins filter below the SiteMinder Web Agent.
6. Restart IIS.

Index

R

S

T

U

V

W