RSA® Access Manager Agent 5.0 SP1 for IBM WebSphere Application Server Installation and Configuration Guide

Contact Information

Go to the RSA corporate web site for regional Customer Support telephone and fax numbers**:** **www.rsa.com**

Trademarks

RSA, the RSA Logo and EMC are either registered trademarks or trademarks of EMC Corporation in the United States and/or other countries. All other trademarks used herein are the property of their respective owners. For a list of EMC trademarks, go to **www.rsa.com/legal/trademarks_list.pdf**.

License agreement

This software and the associated documentation are proprietary and confidential to EMC, are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright notice below. This software and the documentation, and any copies thereof, may not be provided or otherwise made available to any other person.

No title to or ownership of the software or documentation or any intellectual property rights thereto is hereby transferred. Any unauthorized use or reproduction of this software and the documentation may be subject to civil and/or criminal liability. This software is subject to change without notice and should not be construed as a commitment by EMC.

Third-party licenses

This product may include software developed by parties other than RSA. The text of the license agreements applicable to third-party software in this product may be viewed in the **thirdpartylicenses.pdf** file.

Note on encryption technologies

This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when using, importing or exporting this product.

Distribution

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

December 2013

Preface

About This Guide

This guide describes how to install and configure the RSA Access Manager Agent 5.0 SP1 for IBM WebSphere Application Server. It is intended for administrators and other trusted personnel. Do not make this guide available to the general user population.

This guide is intended for network, web site, and security administrators who are responsible for installing and managing the Agent.

This guide assumes that you have a basic understanding of web technologies, UNIX or Windows operating systems, and the WebSphere Application Server environment. For start-to-finish security administration with this Agent, you must have expertise in both WebSphere and the RSA Access Manager administrative tools.

You must be familiar with the use of the WebSphere administrative console, and you must have an understanding of the Custom User Registry. Additionally, you must be familiar with the RSA Access Manager Administrative Console.

RSA Access Manager Agent 5.0 SP1 for IBM WebSphere Application Server Documentation

For more information about RSA Access Manager Agent 5.0, see the following documentation:

***Readme* or *Release Notes*.** Provides information about what is new and changed in this release, as well as workarounds for known issues. The latest version of the *Readme/Release Notes* is available on RSA SecurCare Online at **https://knowledge.rsasecurity.com**.

***Installation and Configuration Guide*.** Describes detailed procedures on how to install and configure RSA Access Manager Agent 5.0.

RSA Access Manager Agent 5.0 Help. Describes day-to-day administration tasks performed in the RSA Security Console. To view Help, click the **Help** tab on the RSA Security Console.

Related Documentation

For more information about products related to RSA Access Manager Agent 5.0, see the following:**RSA Authentication Manager documentation set.** The documentation related to RSA Access Manager is available from RSA SecurCare Online at **https://knowledge.rsasecurity.com.**

To complete the full deployment of the integrated solution for IBM WebSphere 6.0.2, 6.1.0.3, or 7.0, you may require additional resources from the IBM WebSphere documentation. This documentation is available at *IBM Information Center-IBM WebSphere Application Server* web site.

Support and Service

RSA SecurCare Online	https://knowledge.rsasecurity.com
Customer Support Information	www.rsa.com/support
RSA Secured Partner Solutions Directory	www.rsasecured.com

RSA SecurCare Online offers a knowledgebase that contains answers to common questions and solutions to known problems. It also offers information on new releases, important technical news, and software downloads.

The RSA Secured Partner Solutions Directory provides information about third-party hardware and software products that have been certified to work with RSA products. The directory includes Implementation Guides with step-by-step instructions and other information about interoperation of RSA products with these third-party products.

Before You Call Customer Support

Make sure that you have direct access to the computer running the RSA Access Manager Agent software.

Please have the following information available when you call:

- ❑ Your RSA Customer/License ID. You can find this number:

This is a paper license. You can find this number only on the license distribution medium. If you do not have access to the paper-based RSA Customer/License ID, contact RSA Customer Support.

- ❑ RSA Access Manager Agent 5.0 software version number.
- ❑ The make and model of the machine on which the problem occurs.
- ❑ The name and version of the operating system under which the problem occurs.

Before you begin

Directory and File Path Conventions

Directory and file paths are used frequently in this guide. The following conventions are used to describe directory and file locations that may vary according to your individual environment:

- *AGENTBASE:* The directory where the Agent is installed.
- *WASBASE:* The directory where the stand-alone WebSphere is installed.
- *WASPROFILE:* The directory where the WebSphere Profile is created.
- *DMBASE:* The directory where the Deployment Manager is installed.
- *DMPROFILE:* The directory where the Deployment Manager Profile is created.

1 Overview of RSA Access Manager Agent 5.0 SP1

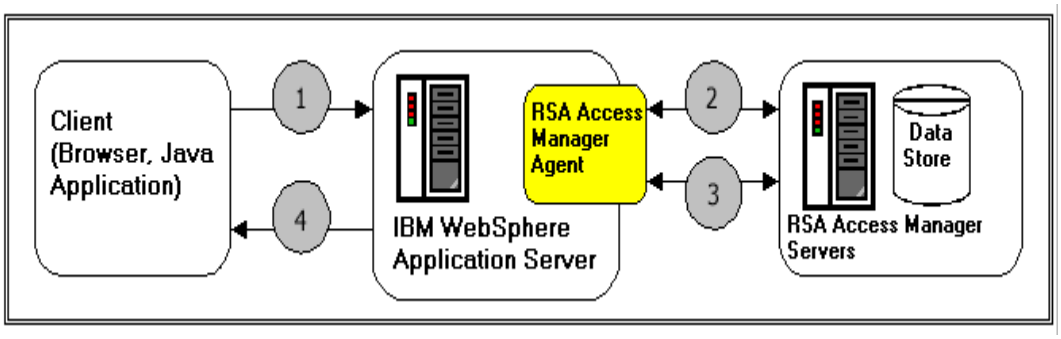
This chapter provides an overview of the Agent installation and configuration process. It describes how the Agent operates in your application server environment, lists system requirements, and summarizes necessary installation steps.

RSA Access Manager Agent 5.0 SP1 for IBM WebSphere Application Server 7.0

The Agent augments WebSphere's native security capabilities. The Agent provides integrated administration of users, groups, role-to-user or role-to-group mapping, and provides runtime support for WebSphere authorization through J2EE resource access checking.

With the Agent deployed, the administrator manages all user, group, and role assignments using the RSA Access Manager Administrative Console. Role-to-user and role-to-group mappings are also done through the RSA Access Manager Administrative Console.

At runtime, the Agent is invoked to authenticate a user when the user tries to access a protected resource. The Agent authenticates the user by verifying the user's credentials with the RSA Access Manager Servers. Once the authentication is successful, the Agent is invoked again to check the user's authorization. The Agent calls the RSA Access Manager Servers to check whether the authenticated user or the user's group has the required role or roles. Access to the J2EE resource is granted based on whether the user or the user's group has the required role or roles. The following graphic illustrates this process at a high level.



1. A client, such as a Java application over Remote Method Invocation (RMI) or an end user's browser over HTTP, makes a request for a protected Servlet, EJB, or method
2. WebSphere invokes the Agent, which performs an authentication check and calls the RSA Access Manager Servers and data store for the user and the group data.
3. After successful authentication, WebSphere invokes the Agent to perform a role-based authorization check. The Agent calls the RSA Access Manager Servers for role-to-user or role-to-group mapping and does the role-based authorization.
4. After authentication and role-based authorization checking, WebSphere returns either the requested resource or an access denial to the client.

Supported Features

This section provides information on the features offered by the Agent.

Support for Linux

The Agent supports Red Hat Enterprise Linux 6 64-bit, SUSE 11 64-bit.

Support for Internationalization

Internationalization is supported for the application agent JSP pages.

Support for multiple editions of WebSphere

The Agent supports WebSphere Application Server Base, and WebSphere Application Server Network Deployment.

Installation with Administrative Security turned on.

The Agent can be installed and reinstalled on a WebSphere that has Administrative Security turned on. The Agent installer accepts the required credentials to connect to WebSphere in order to install the Agent.

Authentication and authorization of enterprise applications

The Agent supports J2EE role-based authentication and authorization of enterprise applications. Both authentication and authorization of resources are managed through RSA Access Manager.

Trust Association Interceptor (TAI)

WebSphere supports authentication using external web servers through Trust Association Interceptors. The Agent provides the RSA Access Manager implementation of TAI, which can be used to trust external web servers that are protected with RSA Access Manager Web Agents.

Java Authorization Contract for Containers (JACC)

The contract enables third-party authorization providers to plug into WebSphere, to make the authorization decisions when a J2EE resource is accessed. The Agent enables the RSA Access Manager implementation of JACC, which can be plugged-into WebSphere to protect the J2EE Resource. This support is only available in the WebSphere version 6.1.0.3 and 7.0.

Authentication and authorization of web services

The Agent supports both authentication and authorization of web services. Web services can be protected at the application level.

Support for IBM WebSphere Portal Server 6.0

The Agent provides SSO and Authentication support for IBM WebSphere Portal Server 6.0 resources.

Multiple authentication types

You can choose the method with which to authenticate users before allowing access to your protected web resources: Basic and Windows NT (user name/password), X.509 Certificate, RSA SecurID (two-factor authentication), and Custom. The Agent supports the use of form-based authentication, which allows you to customize the logon pages displayed to users.

Granular authentication

You can choose whether to base user authentication on more than one authentication type.

SSL

Protects connections between the Agent and RSA Access Manager Servers.

Uniform resource locator (URL) retention

Enables the web server to retain original page request URLs instead of sending users to a default location, such as a home page, after logging on.

Single sign-on (SSO)

After the user has authenticated once, SSO allows the user to access other protected resources without having to reauthenticate each time, provided the user has access privileges to the resource.

Custom user properties

Allows you to publish user information that you have defined as custom user properties in the RSA Access Manager Administrative Console.

On-demand configurable cacheflush

Allows you to flush the Agent cache based on cache type. By passing selected arguments, you can flush combinations of cached protected resources, unprotected resources, user properties, session tokens, "allow" authorizations, and "deny" authorizations.

Improved performance

Authorization Server pool refresh timeouts are now configurable. Also, URL exclusion lists and file extension exclusion lists have been fully implemented. These and other performance settings can be configured in the Agent configuration file, cleartrust.properties.

Proxy and firewall environment support

You can define how cookies are handled by the Agent when proxy servers and firewalls are in use.

User Principal Name (UPN) support

UPN is an Internet-style login name for a user based on the Internet standard RFC 822. The UPN is shorter than the distinguished name and easier to remember. By convention, this should map to the user e-mail id. The value set for this attribute is equal to the length of the user's ID and the domain name. For more information about this attribute, see the Naming Properties topic in the Active Directory guide.

Utilities

The Agent provides several utilities. Ctencrypt allows you to encrypt sensitive Agent configuration information. Synctool allows you to synchronize WebSphere security policies with the RSA Access Manager Servers. Cacheflush allows you to flush the Agent-side cache to affect immediate enforcement of changed RSA Access Manager security policies. CAP tool allows to configure agent in additional profiles.

2 RSA Access Manager Agent 5.0 SP1 Components

The RSA Access Manager Agent for application server is a combination of multiple components.

This chapter explains functionality of the Agent components.

RSA Access Manager Web Filter

Web filters are URL filters that are implemented as ServletFilters. They provide a standard set of authentication and authorization features for web resources like any other RSA Access Manager Agent for web servers. A web filter must be applied to each web application that needs to be protected by RSA Access Manager.

Web filters support different types of authentication. For more information, see "Protecting Web Applications with RSA Access Manager Web Filters" on page 49

WebSphere Custom User Registry

The Agent's functionality is derived from implementing the Custom User Registry provided as part of WebSphere's native security capabilities. The Agent is set up as a WebSphere Custom User Registry to provide the following security services:

- Allow user and group administration through RSA Access Manager
- Provide basic (form and non-form-based) or certificate authentication and password lockout capabilities through RSA Access Manager

RSA Access Manager Authorization Table

This component performs the runtime role-based authorization check for J2EE resources. The roles that are defined in the application must be created in RSA Access Manager as groups, and must be mapped to the users or groups in RSA Access Manager. This process is outlined in Creating Security Policies to Protect J2EE Resources.

RSA Access Manager Java Authorization Contract for Containers (JACC) Provider

The RSA Access Manager Java Authorization Contract for Containers (JACC) provider is a new feature in Websphere Application Server 6.1.0.3. For more information, see the IBM Information Center - WebSphere Application Server 6 Authorization Provider. RSA Access Manager Agent 5.0 SP1 supports this feature. With JACC, authorization is completely delegated to the Agent. This also allows dynamic creation and deletion of resources, groups, and mapping of users and groups to the roles in RSA Access Manager.

The Agent installer automatically configures RSA Access Manager JACC Provider for IBM WebSphere Application Server 6.1.0.3 and 7.0 during the installation. If RSA Access Manager JACC Provider is configured, when you deploy an application it will create corresponding RSA Access Manager groups, applications, and resources for authorization based on RSA Access Manager. Administrators can change the authorization requirements using the RSA Access Manager Administrative Console.

As an example of how RSA Access Manager JACC Provider creates resources and groups in Access Manager, consider a banking application (named BankingApp) with resources protected by a role called "Teller".

RSA Access Manager JACC Provider creates all the protected resources as an Enterprise Application Server Resources in RSA Access Manager. RSA Access Manager JACC Provider then creates a group called "Teller_BankingApp_servername_CTROLE" and grants access permissions to all the protected resources in the BankingApp. If the BankingApp has any user-to-role mappings or group-to-role mappings, then the corresponding users and groups are made members of the "Teller_BankingApp_servername_CTROLE" so that they can get access permission to the resources.

Users and groups can be granted access to the protected resources by the following methods:

- Before deploying the application, specifying users and groups in the Deployment Descriptor.
- During the deployment of the application in the WebSphere Administrative Console, specifying users and groups through the "Security role to user/group mapping" option.
- After deploying the application, through the RSA Access Manager Administrative Console, by adding users and groups to the group that corresponds to the role in the Application.

Security Roles in WebSphere Application Development

When building an application, a developer indicates the WebSphere resources to be protected, and defines security roles controlling access to them. For instance, a banking application requires separate roles for managers and supervisors, in which managers are authorized to transfer money between accounts while supervisors are only allowed to view the accounts. In this case, the manager role is granted access to the J2EE resources that provide the transfer functionality, but the supervisor role is denied this access. This assignment of roles is accomplished through the WebSphere Application Assembly Tool, or directly through the application's Deployment Descriptor.

UnChecked Resources

When building an application, a developer declares a resource as unchecked in the Deployment Descriptor. When a resource is unchecked, the authentication for that resource does not take place and the resource is granted to all users. The resource is treated as an unprotected resource.

Excluded Resources

When building an application, a developer can declare a resource as excluded in the Deployment Descriptor. When a resource is excluded that resource is not granted to any user.

RSA Access Manager Trust Association Interceptor

WebSphere supports using external web servers as front-ends for serving web resources. Requests for web resources are routed to WebSphere through the plug-ins installed in these front-end web servers.

In order for WebSphere to trust these external web servers, the Trust Association Interceptor (TAI) for these front-end web servers must be configured in WebSphere. These TAI modules intercept requests from the front-end web servers and inform WebSphere whether the request has been authenticated by the front-end server or not. WebSphere can delegate authentication to these front-end web servers and provide single sign-on. Once a user has authenticated to the front-end web server, the user does not need to reauthenticate to WebSphere.

RSA provides RSA Access Manager Agents for a number of web servers. These Agents are URL filters that provide a standard set of authentication and authorization features. For a list of WebSphere-supported front-end web servers, refer to the IBM documentation for WebSphere.

WebSphere is bundled with IBM HTTP Server, which can act as a front-end web server. In addition to the Agent for WebSphere, RSA provides an Agent for the IBM HTTP Server. The Agent for WebSphere and the Agent for the IBM HTTP Server may be deployed in the same environment, using the same RSA Access Manager Servers and data store with no need to duplicate user or group data. This allows you to leverage your RSA Access Manager users and groups for authorizing access to J2EE resources with the Agent for WebSphere, while also using that data for full-featured authentication and protection of web server resources with the Agent for the IBM HTTP Server.

Single sign-on from any RSA Access Manager Agent for web servers to the Agent for WebSphere can be done through the RSA Access Manager Trust Association Interceptor. With the appropriate TAI installed, the end users do not need to reauthenticate to the Agent for WebSphere after they have successfully authenticated to an Agent for web servers.

Web Services Filter

Like web applications, web services can also be protected using the web filters provided by RSA Access Manager. Since web services are invoked using the SOAP protocol, not all the features of RSA Access Manager web filter are available. For more information, see Protecting Web Services.

3 Installation Overview

This chapter provides a summary of the platform support and system requirements prior to installation of RSA Access Manager Agent 5.0 SP1.

Make sure that Application Server meets the hardware and software requirements, mentioned in the following sections.

Platform and System Requirements

RSA Access Manager Agent 5.0 SP1 is supported on the following Application Server:

- IBM WebSphere Application Server version 7.0

RSA Access Manager Agent 5.0 SP1 is supported on the following Access Manager Server:

- RSA Access Manager Server 6.2 SP1

RSA Access Manager Agent 5.0 is supported on Red Hat Enterprise Linux -version 6, (64-bit) operating system and SUSE 11 64-bit.

IBM WebSphere Application Server 7.0 is supported on the following platforms:

- SUSE Linux Enterprise Server 11 (64-bit)
- Red Hat Enterprise Linux 6.0 (64-bit)

The Agent must be installed on the same machine as the application server, one Agent for each server instance. A typical installation has the following Agent system requirements:

- 50 MB free disk space, plus another 50 MB free temporary disk space to execute the installer
- 128 MB RAM for the Agent

Installation Overview

The following are the high-level steps for installing this Agent:

1. Set up your WebSphere as described in the IBM documentation.
2. Install your RSA Access Manager Servers before you install this Agent. For more information, see the *RSA Access Manager Servers Installation and Configuration Guide.*

3. Perform preinstallation tasks and collect the information you need to provide during installation of this Agent. For more information, see "Before you begin" on page 19.
4. Run the Agent installation program as described in the Chapter 4, "Installation."
5. Make any other configuration changes in the Agent configuration file as needed. These may include SSL setup tasks or other customized settings listed in Chapter 7, "Configuration and Deployment."
6. Restart WebSphere.
7. Create all the necessary roles and their role-to-user or role-to-group mappings in RSA Access Manager using the "Synctool" on page 103 utility.

4 Installation

This section details how to install and uninstall the Agent. It includes preinstallation tasks, instructions for running the automated installer, guidance for enabling the Custom User Registry options in your WebSphere administrative console, and instructions for uninstalling the Agent.

Before you begin

Before you install the Agent, make sure you have all the information described in the following topics:

- "Prerequisites" on page 19
- "Required Information" on page 19
- "Preinstallation Checklist" on page 21

Prerequisites

Make sure you have met these prerequisites:

- WebSphere Application Server is installed, configured, and running.
- Profile is created in WebSphere.
- The RSA Access Manager Servers are installed and running.
- The RSA Access Manager Administrative Console is installed.
- The RSA Access Manager data store contains a read-write administrator account for the Agent.

Required Information

During installation, you are asked to provide the following information:

- **WebSphere Installation Directory**. The path to your WebSphere server. If the specified location is not a WebSphere installation folder, then the installer throws an error message indicating that the installation path is invalid.
- **WebSphere User Profile Directory**. The path to your running WebSphere's Profile Directory. If the specified location is not a WebSphere User Profile folder, then the installer throws an error message indicating that the path is invalid.

- **WebSphere Administrator User Name and Password**. The installation program sets this user as the WebSphere administrator. This user must be present in the RSA Access Manager datastore, but need not be an administrative user in RSA Access Manager.
- **SOAP Port and HTTP Connector Port.** The SOAP connector port and HTTP Connector Port of the WebSphere Application Server. SOAP Port and HTTP Connector Port details are required to connect to WebSphere Application Server.
- **TrustStore and KeyStore Filename and Passphrase.** Enter TrustStore and LTPA Password. The password used for Light Weight Third Party Authentication. If SSO is desired between multiple WebSphere servers, then all the WebSphere servers should use the same LTPA password.
- **RSA Access Manager Entitlements Server Host and Port.** Allows the Agent to connect to the Entitlements Server. Be prepared to provide the fully qualified hostname or IP address of your Entitlements Server machine as well as the port number it is listening on.
- **RSA Access Manager Entitlements Server Administrative User Name and Password for the Agent**. Allows the Agent to connect to the Entitlements Server as an administrative user using the specified administrative user name and password. This user needs only read-write privileges to the RSA Access Manager data store.
- **RSA Access Manager Entitlements Server Administrative User Name and Password for the Synctool**. This user must have full read-write permission to the RSA Access Manager data store. The synctool uses this account to create all the necessary roles in RSA Access Manager.
- **RSA Access Manager Dispatcher Server(s) or Authorization Server Host and Port.** Allows the Agent to connect to the Dispatcher Server(s) or Authorization Server, which find operational Authorization Servers. Be prepared to provide the fully qualified hostname or IP address as well as the port number it is listening on for each of your Dispatcher Server or Authorization Server machines.
- **RSA Access Manager SSL Mode**. The SSL mode for Agent client connections to the Dispatcher or Authorization and Entitlements Servers. The RSA Access Manager Server and Agent SSL mode settings must match. Check the settings in the parameters cleartrust.net.ssl.use in dispatcher.conf and/or aserver.conf and cleartrust.eserver.api_port.use_ssl in eserver.conf. The RSA Access Manager Servers default value is Authenticated. If your RSA Access Manager Server SSL mode is set to Auth for authenticated SSL connections, you must be prepared to supply the following keystore values to the Agent installer:
  - CA KeyStore Filename
  - CA KeyStore Type
  - CA KeyStore Passphrase
  - Private KeyStore Filename
  - Private KeyStore Type
  - Private KeyStore Passphrase
  - Private Key Alias

– Private Key Passphrase

This information can be found in your RSA Access Manager Servers dispatcher.conf, aserver.conf, and eserver.conf files.

Note: If the RSA Access Manager Server and Agent SSL mode settings do not match, the Agent fails to start. Also, as of RSA Access Manager 6.0, the Administrative Console supports only Clear or Anon SSL connections to the Entitlements Server. Therefore, setting the cleartrust.agent.adapi.ssl parameter to Auth disables the Administrative Console.

Preinstallation Checklist

It may be helpful to complete this checklist and keep it available while installing this Agent. Some of this information is very sensitive from a security standpoint, so be careful about circulating it and destroy it after installation.

Prerequisites:

- WebSphere Application Server is installed, configured, and running.
- Data servers are installed and running with the appropriate RSA Access Manager Data Adapter.
- RSA Access Manager Servers are installed.
- RSA Access Manager Administrative Console is installed.
- A read-write administrator account for the Agent exists in RSA Access Manager Server.

Required Information

WebSphere Information:

- WebSphere Server Installation Directory _____
- WebSphere User Profile Directory _____
- User ID for WebSphere Admin _____
- Password for WebSphere Admin _____
- LTPA Admin Password
- SOAP Port and HTTP Connector Port _____

RSA Access Manager Servers Information:

- Connection mode between Agent and Dispatcher (Clear, Anon, or Auth) _____
- Dispatcher(s) Host (fully qualified hostname or IP address) _____
- Dispatcher(s) Port _____
- Authorization Server(s) (fully qualified hostname or IP address) _____
- Authorization Server(s) Port _____

- Entitlements Server Host _____
- Entitlements Server Port _____
- Connection mode between Agent and Entitlements Server (Clear, Anon, or Auth) _____
- Entitlements Server Admin User Name for Agent _____
- Entitlements Server Admin Password for Agent _____
- Entitlements Server Admin User Name for Synctool _____
- Entitlements Server Admin Password for Synctool

Additional Information:

WebSphere Server Name (server_name parameter in cleartrust.properties) _____

Domain Name for WebSphere Single Sign-on (cookie_domain parameter in cleartrust.properties) _____

If connection mode is set to Auth:

- CA KeyStore Filename _____
- CA KeyStore Type _____
- CA KeyStore Passphrase _____
- Private KeyStore Filename _____
- Private KeyStore Type _____
- Private KeyStore Passphrase _____
- Private Key Alias _____
- Private Key Passphrase _____

5 Installation Modes

The Agent installation program provides different modes for installing the Agent on the required platforms. You need to choose the appropriate installation mode for installing the Agent. Different modes of installing the Agent are described in the following sections.

- GUI-Based Installation
- Console-Based Installation

GUI-Based Installation

The Agent installation program provides a GUI to install the Agent on both Windows and supported UNIX platforms.

To run the installation program on UNIX, you must have X terminal installed and configured.

Console-Based Installation

The Agent installation program provides a console-based interface to install the Agent on supported UNIX platforms. To install in console mode, execute the Setup.bin program with "-i console" as the command line argument. For example: $./Setup.bin -i console.

Installing the Agent

You perform the same installation steps to install the Agent on Windows, Solaris, Linux, and AIX platforms. On Solaris, Linux, and AIX, log on as root user to perform the installation. On Windows, log on as Local Administrator with "Act as part of the operating system" privileges.

Before performing the installation steps, read the following sections for installing in your specific WebSphere environment.

- "Installing on WebSphere Base Edition" on page 23
- "Installing on WebSphere Network Deployment" on page 24
- "Installation Steps" on page 24

Installing on WebSphere Base Edition

To install the Agent on WebSphere base edition, follow the "Installation Steps" on page 24.

Installing on WebSphere Network Deployment

Installing the Agent on a Network Deployment edition is same as on the base edition, except the installer must be run against the Deployment Manager node. That is, the WebSphere installation directory that is chosen during the installation must be the Deployment Manager node. The node agents of all the nodes that are managed by the Deployment Manager must be running during the Agent installation.

Also, the SOAP connector port that is supplied during the Agent installation must be the SOAP connector port of the Deployment Manager.

If any of the nodes goes down during installation and the Agent files get out of sync, manually copy the files to the nodes before starting the server(s) on these nodes. To manually sync the Agent files, follow these steps:

1. Copy the Agent libraries from the DEPLOYMENT MANAGERHOST/lib/ext folder to the /lib/ext folder of the new node.
2. Copy the cleartrust.properties file from the DEPLOYMENT MANAGERHOST/properties folder to the /properties folder of the new node.
3. If the Agent uses certificates, copy all of the certificates to the new node. Also update the cleartrust.properties file copied in step 2 with the certificate path of the new node. Certificates can be kept under the properties folder or the etc folder of the new node. While updating the certificate path, provide the full path of the certificate file. The Agent does not accept relative paths.

Note: When the Agent is installed on a Network Deployment edition, the key store files are automatically copied to the WASPROFILE/properties directory of the remote nodes that are configured. The key store files that are copied are not automatically removed during uninstallation. To remove these files, see Uninstalling the Agent.

Installation Steps

Note: During installation, click **Next** in GUI mode or press ENTER in console mode to proceed to the next panel.

Note: Use WebSphere JDK for successful installation of the Agent.

To install the Agent:

1. Start your WebSphere Application Server.
2. Start the Agent installer.
3. Navigate to the directory where you unzipped the agent package downloaded from RSA website.
4. Double-click **Setup.exe** to begin the installation. The **Welcome** Screen opens. Click **Next**.
5. Ensure that all the prerequisites tasks are met and note down the required information for installing the Agent. Click **Next**.

6. On the License Agreement screen, provide your agreement on the License Agreement and click **Next**.

Note: If you are using putty, change the remote character set to UTF-8. Go to Change settings->translation to change the UTF character set.

7. On the **Installation Type** screen, select **New**.
8. On the Supported Components screen, select the component that protects the application server. The following options are available:

- **Single Sign-On (SSO) and Java Authorization Contract for Containers (JACC)**

  If you want single sign-on with external web servers that are protected with Access Manager application server Agent and want JACC module security provider to be installed, select Single Sign-On (SSO) and Java Authorization Contract for Containers (JACC).

- **Single Sign-On Only**

  If you want Single Sign-On (SSO) with external web servers that are protected with an RSA Access Manager web server Agent, select Single Sign-On Only.

  Note: If you choose **SSO Only**, Trust Association Interceptors (TAI) is used to provide single sign-on with external web servers that are protected with an RSA Access Manager web server Agent. Once the user is authenticated with an RSA Access Manager-protected external web server, the user does not need to be reauthenticated with WebSphere to access resources that are protected with J2EE roles. If you choose **Security Providers** Custom User Registry, Webfilter, Authorization Table, JACC Modules are installed along with the Agent. If you choose **All**, SSO Only and Security Providers modules are installed.

9. (Conditional) If Single Sign-On (SSO) and Java Authorization Contract for Containers is selected, enter password for Light Weight Third Party Authentication (LTPA). Password is used to support SSO between multiple WebSphere Application Server instances. Password must contain 5 to 32 characters.
10. On the **Location** page, enter or browse to the location where you want the Agent to be installed and then click **Next**.
11. On **WebSphere Server Information** page, enter WebSphere Application Server Installation location and profile folder for the selected WebSphere Application Server.
12. On **WebSphere Server Information** page, enter SOAP Port and HTTP Connector Port details to connect to WebSphere Application Server.
13. On **WebSphere Server Information** page, select the **WebSphere Security** checkbox if Global Security is turned on in WebSphere.
14. If Administrative Security is selected, the WebSphere Server Connection Information panels are displayed. On the panels that are displayed, enter the following information:

- WebSphere Administrator User ID
- WebSphere Administrator User Password
- Truststore Filename
- Truststore Passphrase
- Keystore Filename
- Keystore Passphrase

Note: If Administrative Security is selected, WebSphere requires Administrative SOAP clients to use SSL to connect to WebSphere. If SSL configuration of the JMX SOAPConnector PORT is not changed, set:

- Truststore Filename to /etc/DummyClientTrustFile.jks
- Truststore Passphrase to "WebAS"
- Keystore Filename to /etc/DummyClientKeyFile.jks
- Keystore Passphrase to "WebAS"

Note: The SSL configuration of the SOAP Connector port is configured through the sslConfig property under Application Servers/server name/Administration Services/JMX Connectors/SOAPConnector/Custom Properties, in the WebSphere administrative console. If the sslConfig is configured to use a certificate other than the default, you must provide that certificate information for the Truststore and Keystore related inputs. For more information about the certificate related issues, Chapter 17, “Troubleshooting”.

15. The Installation Summary panel is displayed. Verify that the information is correct. The installation begins.

    Note: The installation may take some time depending on your configuration settings.

16. In the Connection Type screen, select the security level for network connection between Access Manager Server and Application Server Agent. There are three types of network connections:
    a. **Clear**. In this type of connection, communication between Web Server Agent and Application Server Agent, is not encrypted.
    b. **Anonymous**. In this type of connection, communication between Web Server Agent and Application Server Agent, is encrypted but does not provide authenticity of the communicating entity.
    c. **Authenticated**. In this type of connection, communication between Web Server Agent and Application Server Agent, is encrypted over SSL with certificate based authentication between them.
17. Enter following details to create **CA KeyStore**:

a. In the **Keystore Format** drop-down menu, select the Keystore file format which holds the Trusted CA certificates.

b. In the **File Name** field, enter the file name that is assigned to the CA keystore file. You can also use the Browse button to browse to the CA Keystore file location.

c. In the **Passphrase** field, enter the passphrase to protect CA keystore.

d. In the C**onfirm Passphrase** field, confirm the entered passphrase.

18. On the **Connection Type** screen, enter the following details to create Private Keystore Key that holds the Private Key details used by RSA Access Manager Servers for selected authentication type:

    a. In the **Keystore Format** drop-down menu, select the Keystore file format.

    b. In the **File Name** field, enter the file name to be assigned to the private keystore file.

    c. In the **Passphrase** field, enter the passphrase to protect private keystore file.

    d. In the **Confirm Passphrase** field, confirm the entered passphrase.

    e. In the **Keystore Alias** field, enter Keystore Alias to be used as the private key by the RSA Access Manager server.

    f. In the **Passphrase** field, enter the passphrase to protect CA keystore.

    g. In the **Confirm Passphrase** field, confirm the entered passphrase.

19. In the **RSA Access Manager Server Information** screen, select the Dispatcher or Authorization Server to connect to Agent and proceed to next screen.

    – Enter the **Server Name/IP** and respective **Port Number**, and click **Add** to enter the list of Dispatcher or Authorization Servers.

20. On the **Enter Entitlements Server Details** panel, provide the following information:

    – Server Name

    – Server Port

    – Admin User Id

    – Admin User Password

21. On the Agent Configuration panel, enter the appropriate WebSphere Server Name and Cookie Domain details.

    Note: In WebSphere 7.0, the domain name given on the Agent Configuration panel is updated as the SSO domain. To verify the domain name, in the WebSphere administrative console, click **Security > Global Security > Web and SIP security > single sign-on (SSO)**. This displays the single sign-on (SSO) properties which include the domain name.

22. Select FIPS if you want to install FIPS compliant RSA Access Manager Servers, else select Non-FIPS.

23. On the **Create Lockbox** panel, enter the **File Location** where you want to store the Lockbox file.
24. Enter the File Name for the Lockbox file. This file will be used to store the Master Password, which is created during the encryption of Configuration Files. File name format must be entered with ".clb" extension.

    Example: LockboxFilename.clb
25. Enter the Lockbox Passphrase to protect Lockbox file. Create a secured passphrase. The passphrase must contain,
    - Minimum 8 characters
    - At least one uppercase
    - At least one lowercase
    - At least one numeric
    - At least one special character

    Note: If you select this option, the crypted utility automatically encrypts the password and accepts this same password as the master startup password needed to reboot the Servers. You can change the password or encrypt more values in your configuration files.
26. Confirm the **Lockbox Passphrase** and click **Next**.
27. The Agent Configuration Summary panel is displayed. Verify that the information is correct, and click **Next** or press ENTER to proceed.
28. After you accept the Agent Configuration Summary information, the Agent configuration files are updated. Upon completion, the Installation Status panel displays the message that the Agent has been installed in the specified location. Based on your success or failure of installation, status is shown as, Congratulations, Failed, or Partially Successful. There are three primary below mentioned conditions due to which installation may be partially successful.
    - LockBox creation failed. For more information, see Lockbox File Tool.
    - Synctool failure. For more information, see Synctool.
    - Application Deployment failed. For more information, see IBM WebSphere documentation.
29. After the installation procedure is complete, click **Done** or press ENTER to exit the installer.
30. Check for the following files in your WASBASE/lib/ext directory:
    - CSPJNI-3.1.jar
    - CSP-3.1.jar
    - LB-3.1.jar
    - LBJNI-3.1.jar
    - axm-admin-api-6.2.jar
    - axm-appagent-common-5.0.jar

- axm-runtime-api-6.2.jar
- cryptojce-6.1.jar
- cryptojcommon-6.1.jar
- log4j-1.2.9.jar
- axm-was-agent-api-5.0.jar
- axm-was-agent-portal-api-5.0.jar

If you have selected FIPS mode during the installation, check for the following files:

- jcm-6.1.jar
- jcmFIPS-6.1.jar

31. Check for the following file in your WebSphere User Profile directory/properties directory: cleartrust.properties, debugenabled.properties, debugdisabled.properties, and log4j.properties.

    Note: If installer is run with SSO+JACC option then, synctool logs are created in AGENTBASE/synctool localtion. For a network deployment if SSO+JACC is chosen then, synctool is not run by the installer. The user to do it manually.

32. Log on to the administrative console.
33. On the navigation tree, click **Security > Global Security**.
34. In the workspace under User account repository, ensure that the Current realm definition is Standalone custom registry and click **Configure**.
35. In the Primary administrative user name field, enter the Websphere Admin User Id and click **Ok**.

    Note: The Websphere Admin User Id must be a valid RSA Access Manager user and must be present in the RSA Access Manager user database.

36. Click **Save** to save the changes to the master configuration.
37. In the workspace under **User account repository**, click **Set as current**.
38. Click **Apply**.
39. Click **Save** to save the changes to the master configuration
40. Restart the WebSphere server.

6 Post-installation Steps

Depending on your requirements, you may need to take additional steps to set up SSL connections and authentication options. These additional steps are described in the Chapter 7, "Configuration and Deployment."

The following sections guide you through verifying the settings that were configured during installation.

Verifying Directories and Files

Verify the directories and files for your particular WebSphere environment:

- Standalone Environment
- Clustered Environment

Standalone Environment

In the standalone environment, verify that the following directories and files have been added.

- AGENTBASE/tools/properties directory:
  - cleartrust.properties
  - debugenabled.properties
  - debugdisabled.properties
  - log4j.properties
  - synctool.properties
- AGENTBASE/jars directory:
  - CSPJNI-3.1.jar
  - CSP-3.1.jar

  - LB-3.1.jar
  - LBJNI-3.1.jar
  - axm-admin-api-6.2.jar
  - axm-appagent-common-5.0.jar
  - axm-runtime-api-6.2.jar
  - cryptojce-6.1.jar
  - cryptojcommon-6.1.jar
  - log4j-1.2.9.jar
  - axm-was-agent-api-5.0.jar
  - axm-was-agent-portal-api-5.0.jar
  - jcm-6.1.jar
  - jcmFIPS-6.1.jar
- AGENTBASE/tools directory:
  - cacheflush.bat (Windows) or cacheflush.sh (UNIX)
  - ctencrypt.bat or ctencrypt.sh
  - synctool.bat or synctool.sh
  - axm-was-agent-cacheflush-ejb-5.0.jar
  - lockbox-tool.bat or lockbox-tool.sh
  - cap.bat or cap.sh
- AGENTBASE/Uninstaller_Data directory
- WASPROFILE/properties directory:
  - cleartrust.properties
  - debugenabled.properties
  - debugdisabled.properties
  - log4j.properties
- WASBASE/lib/ext directory:
  - CSPJNI-3.1.jar
  - CSP-3.1.jar
  - LB-3.1.jar
  - LBJNI-3.1.jar
  - axm-admin-api-6.2.jar
  - axm-appagent-common-5.0.jar
  - axm-runtime-api-6.2.jar
  - cryptojce-6.1.jar

  - cryptojcommon-6.1.jar
  - log4j-1.2.9.jar
  - axm-was-agent-api-5.0.jar
  - axm-was-agent-portal-api-5.0.jar

  If you have selected FIPS mode during the installation, check for the following files:

  - jcm-6.1.jar
  - jcmFIPS-6.1.jar

Clustered Environment

In the clustered environment, verify that the following directories and files have been added.

- AGENTBASE/conf directory:
  - cleartrust.properties
  - debugenabled.properties
  - debugdisabled.properties
  - log4j.properties
  - synctool.properties
- AGENTBASE/jars directory:
  - CSPJNI-3.1.jar
  - CSP-3.1.jar
  - LB-3.1.jar
  - LBJNI-3.1.jar
  - axm-admin-api-6.2.jar
  - axm-appagent-common-5.0.jar
  - axm-runtime-api-6.2.jar
  - cryptojce-6.1.jar
  - cryptojcommon-6.1.jar
  - log4j-1.2.9.jar
  - axm-was-agent-api-5.0.jar
  - axm-was-agent-portal-api-5.0.jar
  - jcm-6.1.jar
  - jcmFIPS-6.1.jar
- AGENTBASE/tools directory:
  - cacheflush.bat (Windows) or cacheflush.sh (UNIX)
  - ctencrypt.bat or ctencrypt.sh

  - synctool.bat or synctool.sh
  - axm-was-agent-cacheflush-ejb-5.0.jar
  - lockbox-tool.bat or lockbox-tool.sh
  - cap.bat or cap.sh
- AGENTBASE/UninstallerData directory
- DMPROFILE/properties directory:
  - cleartrust.properties
  - debugenabled.properties
  - debugdisabled.properties
  - log4j.properties
- DMBASE/lib/ext directory:
  - CSPJNI-3.1.jar
  - CSP-3.1.jar
  - LB-3.1.jar
  - LBJNI-3.1.jar
  - axm-admin-api-6.2.jar
  - axm-appagent-common-5.0.jar
  - axm-runtime-api-6.2.jar
  - cryptojce-6.1.jar
  - cryptojcommon-6.1.jar
  - log4j-1.2.9.jar
  - axm-was-agent-api-5.0.jar
  - axm-was-agent-portal-api-5.0.jar

  If you have selected FIPS mode during the installation, check for the following files:
  - jcm-6.1.jar
  - jcmFIPS-6.1.jar
- WASPROFILE/properties directory (in all the nodes managed by the Deployment Manager):
  - cleartrust.properties
  - debugenabled.properties
  - debugdisabled.properties
  - log4j.properties
  - certificate stores (if the Agent is configured in SSL server authentication mode)

- WASPROFILE/classes directory (in all the nodes managed by the Deployment Manager):
  - CSPJNI-3.1.jar
  - CSP-3.1.jar
  - LB-3.1.jar
  - LBJNI-3.1.jar
  - axm-admin-api-6.2.jar
  - axm-appagent-common-5.0.jar
  - axm-runtime-api-6.2.jar
  - cryptojce-6.1.jar
  - cryptojcommon-6.1.jar
  - log4j-1.2.9.jar
  - axm-was-agent-api-5.0.jar
  - axm-was-agent-portal-api-5.0.jar

  If you have selected FIPS mode during the installation, check for the following files:
  - jcm-6.1.jar
  - jcmFIPS-6.1.jar

Verifying WebSphere Environment

This section lists the configuration changes that the Agent installer makes to WebSphere. To verify these settings, restart WebSphere after installing the Agent and log on to the administrative console.

- "Verifying Custom Registry" on page 35
- "Verifying the Authorization Table" on page 36
- "Verifying Trust Association Interceptor (TAI)" on page 36
- "Verifying Single Sign-On Configuration" on page 36
- "Verifying RSA Access Manager Login Application" on page 36
- "Verifying Cacheflush Application" on page 36
- "Verifying Java Authorization Contract for Containers (JACC) Provider" on page 37

Verifying Custom Registry

To verify custom registry for WebSphere version 7.0:

1. On the navigation tree, click **Security > Global Security.**
2. In the workspace under **User account repository**, ensure that the Current realm definition is **Standalone custom registry**.
3. Select **Standalone custom registry** from Available realm definition and click **Configure**.
4. In the workspace, ensure that the Custom Registry Classname field has a value of com.rsa.cleartrust.websphere.registry.CTRegistry.
5. In the workspace, ensure that the Primary administrative user name, Server user ID, and password have values.

Verifying the Authorization Table

To verify the Authorization Table for WebSphere version 7.0:

1. On the navigation tree, click **Security > Global Security.**
2. In the workspace, click **Custom Properties**.
3. Ensure that the following property with the specified value has been created:

   Property Name: com.ibm.websphere.security.authorizationTable

   Property Value: com.rsa.cleartrust.websphere.authtable.RSAAuthorizationTable

Verifying Trust Association Interceptor (TAI)

To verify TAI for WebSphere version 7.0:

1. On the navigation tree, click **Security > Global Security.**
2. In the workspace, click **Web and SIP security > Trust association**.
3. In the workspace, ensure that the **Enable trust association** checkbox is selected.
4. In the workspace, click **Interceptors** and ensure that the following Interceptor Class Name is displayed:com.ibm.wps.sso.RSATrustAssociationInterceptor.

Verifying Single Sign-On Configuration

To verify Single Sign-On configuration for WebSphere version 7.0:

1. On the navigation tree, click **Security > Global Security**.
2. In the workspace, click **Web and SIP security > Single Sign-On (SSO)**. Ensure that the **Enabled** checkbox is selected.

Verifying RSA Access Manager Login Application

To verify login application for WebSphere version 7.0:

1. On the navigation tree, click **Application > Enterprise Applications.**
2. Ensure that the RSA Access Manager application is installed and started.

Verifying Cacheflush Application

To verify cacheflush application for WebSphere version 7.0:

1. On the navigation tree, click **Application > Application Types > Websphere enterprise application**.
2. Ensure that the axm_cacheflush application is installed and started.

Verifying cleartrust.properties Settings

Verify whether the cleartrust.properties file located in the WASPROFILE/properties directory contains the following properties with appropriate values that are given during the Agent installation:

- cleartrust.agent.ssl.use.
- cleartrust.agent.auth_server_list. Must have a comma-separated list of Authorization Servers. Verify that each Authorization Server is appended with a colon and its port number.

  Note: This Authorization server list is optional and the Dispatch server list is sufficient to obtain the Authorization server list. This **cleartrust.agent.auth_server_list parameter** is not required if **cleartrust.agent.dispatcher_list** is used.

- cleartrust.agent.dispatcher_list. Must have a comma-separated list of dispatcher servers. Verify that each dispatcher server name is appended with a colon and its port number.
- cleartrust.agent.adapi.ssl.
- cleartrust.agent.adapi.host.
- cleartrust.agent.adapi.port.
- cleartrust.agent.adapi.user_id. Must be an encrypted value.
- cleartrust.agent.adapi.user_password. Must be an encrypted value.
- cleartrust.agent.ssl.* If the value of cleartrust.agent.ssl.use or cleartrust.agent.adapi.ssl is "Auth", then the cleartrust.agent.ssl.* properties that end with "passphrase" must be encrypted.
- cleartrust.agent.server_name.
- cleartrust.agent.cookie_domain.
- cleartrust.agent.debug_configuration
- cleartrust.agent.lockbox_file_path

Verifying Java Authorization Contract for Containers (JACC) Provider

To verify authorization contract for containers (JACC) on WebSphere version 7.0:

1. On the navigation tree, click Security > Global Security.
2. In the workspace, click External authorization providers.
3. Ensure that External JACC provider is selected and click Configure.

4. In the workspace, click External JACC provider.
5. Ensure that the following properties have the following specified values:
   - Property Name: Name
   - Property Value: RSA Access Manager
   - Property Name: Policy class name
   - Property Value: com.rsa.cleartrust.websphere.jacc.RSAPolicy
   - Property Name: Policy configuration factory class name
   - Property Value: com.rsa.cleartrust.websphere.jacc.RSAPolicyConfigurationFactory
   - Property Name: Role configuration factory class name
   - Property Value: com.rsa.cleartrust.websphere.jacc.RSARoleConfigurationFactory

Uninstalling the Agent

The following sections contain information on uninstalling the Agent on Windows, Solaris, Linux, and AIX platforms. Before uninstalling read the sections on uninstalling in your specific WebSphere environment.

Uninstalling on WebSphere Network Deployment

When uninstalling the Agent on WebSphere Network Deployment, ensure that all the node agents are running. If not, the Agent certificates and properties files will not be removed from the nodes.

The files must be removed manually in all the nodes that are managed by the Deployment Manager.

Post-Installation Steps for WebSphere Network Deployment

1. Start the applications.
2. Make sure that all the property files are copied to individual nodes. Copy the jars to lib/ext if they are not present.
3. Set the cst path in setupCmdLine script file for individual nodes. For more information, see Step 5 in the section Adding a Node in the Agent-Protected Deployment Manager.
4. Run synctool for the admin node (for SSO +JACC).
5. Restart the complete setup.
6. Rerun synctool (for SSO + JACC).

Important: If the nodes are on different hosts; create lockbox, encrypt the parameters and update the property files accordingly. During uninstall, delete the property files and libraries from individual nodes.

Uninstallation Steps

To uninstall the Agent on Windows, Solaris, AIX, or Linux, execute the shortcut to the Uninstall_WASApplicationAgent50.exe or Uninstall_WASApplicationAgent50.bin, or directly execute the Uninstall_WASApplicationAgent50.exe or Uninstall_WASApplicationAgent50.bin under *Agent installation directory/UninstallerData*.

To uninstall the Agent on Windows, Solaris, AIX, or Linux:

1. Ensure that WebSphere server is running.
2. Go to the Uninstaller_Data directory under the Agent installation directory.
3. Execute Uninstall_WASApplicationAgent50.bin on UNIX or Uninstall_WASApplicationAgent50.exe on Windows.
4. In GUI mode, this opens an uninstaller window. Click **Uninstall** in the lower-right corner of the panel. On the window that is displayed, enter the following information if **Administrative Security** is selected.

- WebSphere Admin User ID
- WebSphere Admin password
- WebSphere SOAP Port number
- WebSphere Truststore Filename
- WebSphere Truststore Passphrase
- WebSphere Keystore Filename
- WebSphere Keystore Passphrase
- WebSphere Application Server and Profile Location

Note: For more information about certificate-related issues, see Chapter 17, "Troubleshooting."

If you are doing a console mode uninstallation on UNIX, proceed with uninstallation by providing the previous information and pressing ENTER.

5. After you provide the appropriate information, the uninstaller unconfigures the Agent. For network deployment, uninstaller prompts you for permission to issue stop requests to stop the network deployment setup.
6. The uninstaller displays the status of the uninstallation process.
7. Restart the WebSphere server, since it is automatically stopped by the uninstaller during uninstallation.

Note: Lockbox in AgentBase will not be deleted by the installer.

Note: If you have performed an SSO-only installation, the Agent uninstaller disables WebSphere SSO and TAI configurations except Global Security settings on WebSphere. Also, the Agent uninstaller removes the applications installed on WebSphere. If your WebSphere deployment requires SSO, you must manually re-enable it after uninstalling the Agent.

Note: If you have performed a complete Agent installation, the Agent uninstaller disables WebSphere SSO, TAI configurations, JACC, CustomRegistry, and Global Security settings. Also, the Agent uninstaller removes the applications installed on WebSphere. To re-enable Global Security and other settings required for fresh Agent installation, perform the following:Open the security.xml located in the WASBASE/profiles/WASPROFILE/config/cells/CELL_NAME/ directory. Replace the content in security.xml with the content of security.xml PRECT file located in the same folder.Restart the WebSphere application server.

7 Configuration and Deployment

This section describes your options for configuring the Agent. It also provides limited guidance and high-level process flow examples for setting up policies to protect J2EE resources in the RSA Access Manager and WebSphere integrated solution. Complete details on the configuration of WebSphere are provided in IBM's WebSphere documentation. This documentation is available at **http://www.ibm.com/software/webservers/appserv/was.**

Configuring the Agent

All of the Agent configuration parameters are stored in a text file, cleartrust.properties, which is read at startup by the system components. The configuration file provides parameter descriptions, allowed values, and examples. This file is located in the properties directory under the WebSphere User Profile folder.

Though a standard installation automatically populates the required values of cleartrust.properties for deployment, you can edit this file to change the configured properties to meet your deployment requirements. Each time you make changes to cleartrust.properties, you must restart WebSphere in order to properly load the new settings.

Parameter Settings

The parameters listed in this section must be set during installation in order for the Agent to start and operate successfully:

- cleartrust.agent.ssl.use. This property specifies the mode of connections to the RSA Access Manager Servers. The allowed values for this property are Clear, Anon, and Auth. This setting must match the settings for cleartrust.net.ssl.use in your aserver.conf and dispatcher.conf files.

For authenticated SSL connections, you must configure additional parameters as described in "Setting Up the Agent in Authenticated SSL Mode" on page 48cleartrust.agent.dispatcher_list. This is a comma-separated list of RSA Access Manager Dispatcher Servers. Each Dispatcher Server is expressed in the format hostname:listener_port or ipaddress:listener_port. The default listener port for the Dispatcher Server is 5608.

- cleartrust.agent.adapi.host. This property specifies the IP address (or hostname) of the RSA Access Manager Entitlements Server.
- cleartrust.agent.adapi.port. This property specifies the port number of the Entitlements Server. The default value is 5601.
- cleartrust.agent.adapi.ssl. This property specifies the mode of connection to make to the Entitlements Server. The allowed values for this property are Clear, Anon, and Auth. This parameter and the value for the property cleartrust.eserver.api_port.use_ssl in the eserver.conf file of the Entitlements Server must match.

Note: If you select Auth SSL mode in either or both of these settings, you must provide the keystore information listed in "Required Information" on page 19 and you must provide a valid keystore for the WebSphere server. For more information, see "Providing Keystore Files for the Agent" on page 48 the Administrative Console supports only Clear or Anon SSL connections to the Entitlements Server.

For authenticated SSL connections, you must configure additional parameters as described in "Setting Up the Agent in Authenticated SSL Mode" on page 48.

- cleartrust.agent.adapi.user_id. This parameter specifies the Entitlements Server administrator's user name. The value of this property is encrypted. To create an encrypted user name, see "Encrypted Parameters" on page 43.
- cleartrust.agent.adapi.user_password. This parameter specifies the Entitlements Server administrator's password. The value for this parameter is encrypted. To create an encrypted password, see "Encrypted Parameters" on page 43.
- cleartrust.agent.SECURITYBLOCK. This random string is used to encrypt the sensitive parameters specified in the cleartrust.properties file. This parameter is also used by the WebSphere Agent ServletFilter to encrypt the URL retention cookie. For more details about how to generate a secure random string, see "Encrypted Parameters" on page 43.

Additional Parameters:

- cleartrust.agent.max_open_connections. This parameter specifies the maximum number of open server connections that the Agent must maintain for a server pool.
- cleartrust.agent.rtapi.retry_count. This parameter specifies the number of times the Runtime API must attempt to connect to an RSA Access Manager Authorization Server before deciding whether the server is dead or not.

- cleartrust.agent.auth_server_mode. This parameter specifies the mode by which the Agent connection pools route outgoing requests to the Authorization Servers. If set to STANDARD, all requests are routed to a single Authorization Server until a failure occurs. If a failure occurs, requests are routed to the next server in the list. If set to DISTRIBUTED, requests are routed in a sequential style to all available Authorization Servers.
- cleartrust.agent.adapi.admin_group. This parameter specifies the administrative group of the Entitlements Server administrative user.
- cleartrust.agent.adapi.admin_role. This parameter specifies the administrative role of the Entitlements Server administrative user.

Encrypted Parameters

Sensitive information in the configuration file must be encrypted in order for the Agent and its tools to work properly. The Agent installer encrypts all the sensitive properties during installation. It is also possible to encrypt this information manually using the ctencrypt tool that is provided with the Agent. This command line tool takes input in the form of name=value pairs and displays the output as name=encrypted value pairs.

The sensitive information is encrypted with the help of a random block called SECURITYBLOCK. The SECURITYBLOCK that is used to encrypt the properties in the properties file must be present in the properties file, for the Agent to decrypt the encrypted values. The ctencrypt utility can optionally take SECURITYBLOCK as an input value and use it to encrypt the other parameters that are passed. If SECURITYBLOCK is not passed as the input, the ctencrypt utility generates a random SECURITYBLOCK and displays it on the screen.

Sensitive parameters that need to be encrypted are:

cleartrust.properties

- cleartrust.agent.adapi.user_id
- cleartrust.agent.adapi.user_password
- cleartrust.agent.ssl.ca.keystore_passphrase
- cleartrust.agent.ssl.private.keystore_passphrase
- cleartrust.agent.ssl.private.key_passphrase

sycntool.properties

- AdminUserName
- AdminUserPass
- CAkeystore_passphrase
- PrivateKeystore_passphrase
- WebSphereServerSecurityUserName
- WebSphereServerSecurityUserPass
- WebSphereTrustStore_passphrase
- WebSphereKeyStore_passphrase

For examples, see “Ctencrypt” on page 109.

Server Pool Settings

These settings provide information about the RSA Access Manager Dispatcher and Authorization Servers, and define how the Agent communicates with these Servers. Most of the relevant parameters in cleartrust.properties begin with .auth_server and .dispatcher.

While it is possible to run an Agent with only one Dispatcher or Authorization Server in the pool, typical deployment scenarios demand that the Agent create a pool of multiple Authorization Servers. These connection pools are managed either "dynamically" by the Dispatcher, or "statically" according to lists that you define in the cleartrust.properties parameters .dispatcher_list and .auth_server_list.

When dynamic and static lists are used together in creating connection pools, dynamically managed Servers are sorted first. Static Servers are sorted in the order that you have listed them in .auth_server_list.

Note: Static Servers must be listed in IP address format, not hostname format, to interoperate with dynamic lists as described here. The Dispatcher identifies all dynamic Servers as IP addresses and performs a string comparison to the list of static Servers.

If the RSA Access Manager environment is using multiple Authorization Servers, the KeyClient.sec entries for each Authorization Server must be listed in the KeyServer.sec file on all key server machines. Older RSA Access Manager Agents obtain keys directly from the Key Server. In RSA Access Manager Agent 5.0 SP1, the Agent obtains keys from the Authorization Server. For details, see the *RSA Access Manager Servers Installation and Configuration Guide*.

The behavior of connection pools is further controlled by these settings:

- “Location Class and Priority” on page 44
- “Standard or Distributed Connection Mode” on page 46
- “Pool Refresh” on page 46

Location Class and Priority

You can configure the Agent to use a preferred set of Authorization Servers. This allows the Agent to use an Authorization Server in the same geographic location, minimizing the response time for user requests.

You have configuration options to specify a specific location class for each Authorization Server and RSA Access Manager-protected server. In each Agent configuration file you can specify preferred location classes by setting the .location_class_priority parameter. Servers without a class or belonging to a class not present in the location class preference list are placed in a separate, implicitly defined class of the lowest priority.

Servers within a connection pool are ordered and sorted by these settings for location class and priority. Within each class, dynamic Servers (those returned from Dispatchers) are sorted before static Servers. Static Servers are sorted in the order that they are listed in the .auth_server_list parameter. Dynamic Servers within the same priority class may be inserted in random order to prevent multiple Agents in standard mode from all using the same Server.

To configure location class and priority settings:

1. Install and configure all participating Agents and Servers.
2. Configure all grouped Authorization Servers with the same location class, as defined in aserver.conf by the .location_class parameter. Use the same location class name in all participating Agent and Server configuration files.
3. In the cleartrust.properties file for each Agent, specify the location class that the Agent must use. If you are using more than one location class, set the priority using .location_class_priority as shown in this example:

   ```
   cleartrust.agent.location_class_priority=NY,LA,UK
   ```
4. Restart Agents and Authorization Servers for the settings to take effect.

Important: The name you assign for the location class must be consistently used in all participating Agent and Server configuration files. Assigning location classes to Servers does not affect pooling in any way until you assign them a priority in cleartrust.properties.

Location Class and Priority Settings: Example

In this example, dN are dynamic Servers, sN are static Servers and A,B,C,… are location classes. Therefore, s3:F is static Server number three in class F.

The location class parameter sets priority as: B, C

Dispatcher returns Servers: d1:A, d2:B, d3:C, d4

Statically added Servers: s1:A, s2:B, s3:C, s4:C

Resulting list: d2:B, s2:B, d3:C, s3:C, s4:C, {d4, d1:A}, s1:A

The Agent opens connections to Servers in class B first, dynamic ones before static ones, then Servers in class C, and lastly any other classes. Dynamic Servers of the same priority can be sorted in any order (indicated by braces).

Connections are opened to all Servers in the list, unless the max_open_connections parameter has been set to a positive value, in which case at most that many connections are created, starting with the first Server in the list and stopping when a sufficient number of connections have been established.

Behavior with max_open_connections defined: Example

Using the same notation as the previous example, this example indicates Servers with open connections as underlined. It is assumed that all Servers can be successfully contacted.

The max_open_connections parameter is set to 3

Resulting list: d2:B, s2:B, d3:C, s3:C, s4:C, {d4, d1:A}, s1:A

Or, if s2 and d3 are unavailable:

List: d2:B, s2:B, d3:C, s3:C, s4:C, {d4, d1:A}, s1:A

If the same Server appears in both a static and a dynamic list, the dynamic entry is ignored. Also, if a static Server is added multiple times, only the first occurrence is used. Servers are considered identical if they have the same port number, the same hostname, and they belong to a class of the same priority. All classes not present in the priority list are treated as equal.

Standard or Distributed Connection Mode

Connection mode may be Standard or Distributed. In Standard mode, the Agent sends requests to the first available Server in the list. In Distributed mode, the Agent distributes requests among all available Servers, static and dynamic, in the first priority class with open connections.

An Agent, operating in distributed mode, chooses among the Servers in a given priority class the Server with the least number of outstanding packets. When all servers are equal in outstanding packets, it picks the Server with the lowest average response time. Under low loads, a fraction of the requests are distributed randomly among eligible Servers in order to keep the response time estimates updated and to select faster Servers.

Authorization Server connection mode is defined in the Authorization Server configuration file, aserver.conf. For more information, see the *RSA Access Manager Servers Installation and Configuration Guide* and the commenting in aserver.conf.

Pool Refresh

The only time the Agent opens new connections is during pool refresh. Pools are implicitly refreshed upon creation. Agents may explicitly refresh the pool at any time. RSA recommends forcing pool refreshes periodically, because it is the only time changes in Dispatcher Servers lists can be detected.

Pool refresh is a thread-safe operation that does not interfere with threads using existing open Server connections. Connections are not unnecessarily closed, nor are other threads blocked from using connections during the pool refresh.

If the pool has a Dispatcher list, Dispatchers are tried in the order listed, always starting with the first one and continuing until reaching one that supplies a non-empty list of Authorization Servers, of which at least one can be successfully contacted. Whether the Dispatcher connection is kept open between pool refreshes is undefined.

The Server list of the pool is updated with any changes received from the Dispatcher. New Servers are inserted into the list, ordered by class, and Servers no longer present in the Dispatcher list are closed and removed. Dynamic Servers already present in the list should not be re-randomized, to avoid unnecessary opening and closing of connections when max_open_connections limits the number of open connections to part of the class. New dynamic Servers are inserted in a random position among the dynamic Servers of the same class.

When there have been no failed connections since the last pool refresh, and the Dispatcher returns the same list of Servers as the last time, though possibly in a different order, no Server connections are created or closed. This is the normal case in a stable production environment. If none of the dynamic Servers returned from a Dispatcher can be successfully contacted, that list is ignored, and any existing open connections to dynamic Servers are not closed. Any existing connections in the pool are verified by sending a test request. If a connection is determined inoperable, it is closed and eligible to be reopened.

Additional Settings

General information is provided here for many of the Agent features. More detailed information is provided for some of the major features. Read the Agent configuration file to familiarize yourself with all of the individual parameters.

Cookie handling. Define cookie lifetime, idle time, expiration, and so on. These are standard cookie handling parameters. For more information, see cleartrust.properties. To achieve cookie compatibility and SSO with web servers running older versions of the RSA Access Manager Agent, see “Handling Backward Cookie Compatibility” on page 77

Proxy settings. Define how to handle web servers that reside behind a proxy server. See Chapter 9, “Configuring Proxy Environments.”

User properties. Define whether to allow custom user properties defined in the RSA Access Manager Administrative Console to be published in HTTP headers.

Debug settings. Agent can be configured to print debug information in the SystemOut.log file. For more information, refer to the cleartrust.agent.trace_enabled parameter in cleartrust.properties.

Authentication settings. Define the type of authentication to be used: Basic, Windows NT, Certificate, RSA SecurID, and Custom. The default is Basic. See Chapter 7, “Configuration and Deployment.”.

URL retention settings. Define how original URLs must be retained when users are redirected to the logon forms for authentication. See “Implementing URL Retention” on page 78.

URL and file extension exclusion settings. Define which URLs and file extensions are to be excluded from access control checks. By default, image files (.gif and .jpg) are excluded from access control checks to enhance Agent performance. See .url_exclusion_list and .extension_exclusion_list in cleartrust.properties.

Cache settings. Configuring cache settings can help reduce the time between when a user requests a protected resource and when that resource is accessed. See Chapter 16, “Caching and Performance Tuning.”

SSO. Determine whether to use single sign-on to allow users, once authenticated, to access additional protected resources without reauthenticating. See Chapter 12, “Configuring SSO.”

Logon information. Define the location of your end-user authentication logon forms: initial logon pages, logon error message pages, inactive or expired account message pages, user lockout pages, access denied pages, server error pages, and invalid certificate message pages. Look for *_location.

Setting Up the Agent in Authenticated SSL Mode

If your RSA Access Manager Entitlements and Authorization/Dispatcher Servers are configured for authenticated SSL connections with clients, then the Agent must also be configured to use authenticated SSL. The mode of these SSL connections in the RSA Access Manager Servers is governed by these parameters:

- cleartrust.net.ssl.use in aserver.conf
- cleartrust.net.ssl.use in dispatcher.conf
- cleartrust.net.ssl.use in eserver.conf
- cleartrust.eserver.api_port.use_ssl in eserver.conf for admin APIs only

Make sure that you check these settings before setting up the Agent in authenticated SSL mode. Authenticated SSL setup on the Agent requires you to provide valid keystore and CA keystore files, and to set all relevant SSL parameters in cleartrust.properties. These procedures are detailed in the following sections:

- "Providing Keystore Files for the Agent" on page 48
- "Setting SSL Parameters" on page 48

Providing Keystore Files for the Agent

To enable authenticated SSL between the Agent and the RSA Access Manager Servers, you must generate a private keystore file and a CA keystore file for the Agent, or copy valid existing keystore files into a directory.

If you prefer to generate keystore files for the Agent, use the certool utility provided with your RSA Access Manager Servers installation. This tool interoperates with RSA Keon to generate certificates and keystores used for configuring Intercomponent Security in the RSA Access Manager Servers. Guidance for using this certool utility is provided in the Servers Installation and Configuration Guide.

Alternately, you may copy the existing keystore files from the conf directory of your RSA Access Manager Servers. The Agent can use the same CA keystore and private keystore files as a Server.

Setting SSL Parameters

The standard installation prompts you to set all SSL parameters if you select Authenticated (Auth) as your SSL mode. If you have opted for manual configuration, you must set these parameters to enable authenticated SSL:

cleartrust.agent.ssl.ca.keystore_file. The filename that contains trusted CA certificates. This name must be an absolute path to the filename.

cleartrust.agent.ssl.ca.keystore_type. The type of CA keystore, either JKS or PKCS12. This parameter defaults to PKCS 12 if the parameter is commented out.

cleartrust.agent.ssl.ca.keystore_provider. The provider of the keystore algorithm used for unlocking and using the CA keystore. This parameter defaults to the RSA PKCS#12 provider if the parameter is commented out. If ca.keystore_type is JKS, you must set this parameter to SUN.

cleartrust.agent.ssl.ca.keystore_passphrase. The password to unlock the CA keystore. Passwords are case sensitive. The value of this parameter must be encrypted. For details about encrypted parameters, see "Encrypted Parameters" on page 43.

cleartrust.agent.ssl.private.keystore_file. The filename that contains the private key and corresponding certificate used to establish authenticated SSL connections. This name must be a filename with absolute path.

cleartrust.agent.ssl.private.keystore_type. The type of private keystore, either JKS or PKCS 12. This parameter defaults to PKCS 12 if the parameter is commented out.

cleartrust.agent.ssl.private.keystore_provider. The provider of the keystore algorithm used for unlocking and using the private keystore. This parameter defaults to the RSA PKCS#12 provider if the parameter is commented out. If ca.keystore_type is JKS, you must set this parameter to SUN.

cleartrust.agent.ssl.private.keystore_passphrase. The password to unlock the private keystore. Passwords are case sensitive. The value of this parameter must be encrypted. For details about encrypted parameters, see "Encrypted Parameters" on page 43.

cleartrust.agent.ssl.private.key_alias. The alias of the private key present in the private keystore.

cleartrust.agent.ssl.private.key_passphrase. The password to unlock the private key specified by the cleartrust.agent.ssl.private.key_alias in the private keystore. Passwords are case sensitive. The value of this parameter must be encrypted. See "Encrypted Parameters" on page 43.

Protecting Web Applications with RSA Access Manager Web Filters

The RSA Access Manager web filters provide enhanced authentication and authorization support for web applications. The protected resources are centrally managed through the RSA Access Manager Administrative Console. The advantage of web filter-based protection of web applications as opposed to the J2EE role-based protection is that the web filters support more authentication types, such as:

- Basic (form and non-form-based)
- Windows NT (form and non-form-based)
- RSA SecurID (form-based)
- Custom (form-based)
- Certificate

Web filters are characterized by the following features:

- URL exclusion
- Extension exclusion

- Multiple authentication support
- Cookie exclusion support
- Cookie IP check
- Cookie IP check exclusion support
- Enhanced caching support (security parameters only)

Setting Up Authentication Using Web Filters

Authentication verifies user credentials before allowing a user to access a protected resource. RSA Access Manager provides its own internal authentication type by validating the user ID and password against the RSA Access Manager data store. RSA Access Manager also interoperates with several external authentication types, such as Microsoft NT PDC and X.509 certificates. For added security, you can combine various authentication types for specific resources.

Basic, Windows NT, and RSA SecurID authentication types are handled by the RSA Access Manager Authorization Servers and are logged by the RSA Access Manager system according to the logging preferences you define when configuring your RSA Access Manager Servers. The RSA Access Manager Runtime API can be used to extend the functionality of these authentication types in the Authorization Server. For more information, see the *RSA Access Manager Developer's Guide*.

Custom authentication (using the Authorization Server) and Certificate authentication are performed at the Agent.

The authentication types use logon forms to present web pages to the user for authentication. You can customize the forms so that they have the same look and feel as your other company web pages. The .form_based_enabled parameter in cleartrust.properties tells the Agent to present web pages to the user, while the .login_* parameters specify the location of the web pages to be presented.

Configuring Applications for Authentication

To configure applications for authentication:

1. Add the following lines to the web.xml file under the WEB-INF folder in the web application that you want to protect with an RSA Access Manager web filter.

```
<filter>
<filter-name>CTLoginFilter</filter-name>
<filter-class>com.rsa.cleartrust.websphere.CTLoginFilter</fi
lter-class>
</filter>
<filter-mapping>
<filter-name>CTLoginFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

You can modify the web.xml file manually or use the Application Assembly Tool to add filter and filter-mapping sections into the web.xml file.

2. Add the Websphere server as an Enhanced App Server in RSA Access Manager.

3. Protect the URL of the web application in RSA Access Manager. In order to protect all the web resources under the web application, add the entire context root of that web application as protected resources in RSA Access Manager. While adding the resource to RSA Access Manager, select the Resource Type as **Enhanced App Server Resource (J2EE)** and the J2EE Type as **Web Resource (JSP, HTML, Java servlet, static web resources**).
4. Install the RSA Access Manager web filter-enabled application into the application server.

Note: If you want to enable the RSA Access Manager web filter to protect an installed application without reinstalling the application, you can directly modify the web.xml file under the *WASPROFILE/config/cells/CELL_NAME/applications/APPLICATION_EAR_NAME/deployments/APPLICATION_NAME/WEB_APPLICATION/WEB-INF/* folder. After modifying the web.xml file, stop and restart the application.

If the application is installed in a Deployment Manager, the web.xml file must be modified under the Deployment Manager, and all the nodes must be synchronized again.

Configuring Authentication Types

Authentication types are defined in cleartrust.agent.auth_resource_list in cleartrust.properties. The cleartrust.agent.default_auth_mode parameter defines which authentication type to use if none has been defined in the Auth resource list for a protected resource. The default for these parameters is Basic.

RSA Access Manager provides support for the following authentication types:

- "Basic Authentication" on page 51
- "Windows NT Authentication" on page 52
- "NTLM Authentication" on page 52
- "RSA SecurID Authentication" on page 53
- "Custom Authentication" on page 53
- "Certificate Authentication" on page 54

In all cases, the configured authentication type prompts the user attempting to access an RSA Access Manager-protected resource to provide the appropriate identification credentials. If the authentication type accepts the credentials, RSA Access Manager then checks for the user's authorization privileges on the requested resource.

Basic Authentication

Basic authentication validates the user ID and password provided at logon with the user account information in the RSA Access Manager data store. This is the default authentication type for all RSA Access Manager-protected resources.

See these parameters in cleartrust.properties: cleartrust.agent.auth.resource_list, cleartrust.agent.default_auth_mode, cleartrust.agent.form_based_enabled, and cleartrust.agent.login_*.

Windows NT Authentication

This authentication type supports authenticating users against their Windows NT domain accounts. For each user, an account with the same Windows NT user name must also exist in the RSA Access Manager data store.

This type also requires that you set this value in the aserver.conf file on your RSA Access Manager Authorization Server in the following format:

```
cleartrust.aserver.nt_domain_controllers=PDC,BDC
```

Set this parameter to a comma-separated list of NetBIOS names of your Windows NT primary domain controllers (PDC) and backup domain controllers (BDC). Restart the Authorization Server.

If user ids are not guaranteed to be present in all domains, then the mapping between the user ids can be done using the following parameter in the aserver.conf file.

cleartrust.aserver.authn.ntlm_name_mapping=true

See these parameters in cleartrust.properties: cleartrust.agent.auth.resource_list, cleartrust.agent.default_auth_mode, cleartrust.agent.form_based_enabled, and cleartrust.agent.login_*.

NTLM Authentication

This authentication type supports authenticating users against their Windows NT domain accounts. This type also requires that you set this value in the aserver.conf file on your RSA Access Manager Authorization Server in the following format:

```
cleartrust.aserver.ntlm_domain_controllers=PDC,BDC
```

Set this parameter to a comma-separated list of NetBIOS names of your Windows NT primary domain controllers (PDC) and backup domain controllers (BDC).

If desired user ids does not present in both NT domain and RSA Access Manager datastore, set the following parameters in the aserver.conf file:

```
cleartrust.aserver.authn.ntlm_name_mapping=true
```

For more information on how to run the mapping tool, see the *RSA Access Manager Servers Installation and Configuration Guide*.

See these parameters in cleartrust.properties: cleartrust.agent.auth.resource_list, cleartrust.agent.default_auth_mode, cleartrust.agent.form_based_enabled, and cleartrust.agent.login_*.

Note: If you are using IBM WebSphere Application Server 6.0, you need to upgrade to IBM WebSphere Application Server 6.1.0.3 to setup NTLM authentication.

RSA SecurID Authentication

This authentication type supports RSA SecurID two-factor authentication to validate a user's user name and passcode at logon against the credentials stored in the RSA Authentication Manager. For each user, an account with the same RSA Authentication Manager user name must also exist in the RSA Access Manager data store. A passcode is a combination of a user's PIN and RSA SecurID valid tokencode entered as one continuous string. If the passcode is valid, the RSA Authentication Manager returns the request to the RSA Access Manager Authorization Server for access control checking.

See these parameters in cleartrust.properties: cleartrust.agent.auth.resource_list, cleartrust.agent.default_auth_mode, cleartrust.agent.form_based_enabled, and cleartrust.agent.login_*.

Custom Authentication

You can use the RSA Access Manager Runtime API to create your own customized authentication routine and create your own error messages and logging.

Setting Up Custom Authentication

In order to use your own customized authentication routine, you need to configure both the Agent and the RSA Access Manager Authorization Server to accept the new Custom authentication type.

To set up custom authentication:

1. Define your Custom authentication type by implementing the Authenticator interface provided in the Java Runtime API included in the RSA Access Manager Server package.
2. Build your Custom authentication Java file and create a JAR file.
3. Place your Custom authentication JAR file in the RSA Access Manager Server's /lib directory.
4. Specify your Custom authentication file in the aserver.conf parameter, cleartrust.aserver.customauth.classes.
5. Specify your Custom authentication class name in the Agent parameter .custom_auth. The name must exactly match the value returned by the method getAuthenticationType in the Custom authentication class. For example:

   ```
   cleartrust.aserver.customauth.classes=sirrus.runtime.customauth.SampleAuth
   ```

Important: You must list multiple Custom authentication classes and names in the same order in the corresponding Server and Agent list parameters. Values in the aserver.conf parameter cleartrust.aserver.customauth.classes and the parameter cleartrust.agent.custom_auth in cleartrust.properties must match each other exactly.

6. Specify the resources that require Custom authentication in the Agent parameter cleartrust.agent.auth_resource_list, using the Custom authentication name. See the example in step 7.

7. Specify the location of form pages for Custom authentication, as shown in this example configuration for two types named "MyAuth" and "SampleAuth":

```
cleartrust.agent.custom_auth=MyAuth,SampleAuth
cleartrust.agent.auth_resource_list=/page1.html=MyAuth,/page
2.html=SampleAuth
cleartrust.agent.login_form_location_custom1=/ctappagent/ct_
logon_custom1.html
cleartrust.agent.login_form_location_custom2=/ctappagent/ct_
logon_custom2.html
```

For details on using the Authenticator interface, see the *RSA Access Manager API Javadocs*.

See these parameters in cleartrust.properties: cleartrust.agent.custom_auth, cleartrust.agent.auth.resource_list, cleartrust.agent.default_auth_mode, cleartrust.agent.form_based_enabled, and cleartrust.agent.login_*.

Certificate Authentication

Certificate authentication supports X.509 certificates for authentication, such as those issued by the RSA Keon Certificate Authority. Setting up Certificate authentication varies depending on the certificate authority (CA) and application server you are using. See the *Application Server* documentation.

Note: Whenever a user's certificate DN is entered into RSA Access Manager, make sure that the DN entry matches what is returned by the application server (or web server if the application server is front-ended by a web server). For example: *CN=myName, OU=myOrgUnit, O=myOrg, L=myLocality, ST=myState, C=myCountry.*

Prerequisites

You must have an identity and CA root certificate installed on your WebSphere server. For details, see the WebSphere documentation.

To set up Certificate authentication:

1. Configure the WebSphere Application Server to accept HTTPS requests from client browsers or applications. The application server must be configured to require client certificates. for more details about configuration, refer to the IBM WebSphere documentation.
2. Define which resources require Certificate authentication, and specify CERTIFICATE for those resources in cleartrust.properties. See cleartrust.agent.auth_resource_list and cleartrust.agent.default_auth_mode.
3. Configure your client machines for certificates. The user must be issued a browser certificate or personal certificate from the CA, which must then be installed and configured in the browser. For details, see your web browser documentation.

- The distinguished name (DN) of the client-side certificate must match the DN in the user's account in the RSA Access Manager data store. The cleartrust.data.ldap.user.attributemap.certdn: dn parameter in ldap.conf specifies the LDAP attribute that stores the user's certificate DN.

- If the user entry DN in the LDAP directory is not the same as the DN of the user's browser certificate, change the LDAP attribute from DN to ctscUserDN, or to another modifiable attribute in your user object class. In Active Directory, in order for Certificate authentication to work, the selected or added attribute must have a syntax type of "Unicode String" or "Directory String".
- **For Microsoft Active Directory:** Note that the DN value specified absolutely must be valid within the DIT, due to the directory's active maintenance of referential integrity. According to Microsoft documentation, "If the referenced object is renamed or moved, Active Directory ensures that the attribute reflects the change. If the attribute is reset with a new DN, the attribute is referenced to the object represented by the new DN." For more information, see the DN mapping parameters in the Servers Installation and Configuration Guide.

Note: If the certificate DN contains the uid attribute, certain browsers convert uid to the object id number "0.9.2342.19200300.100.1.1". For example, if the user's DN is "uid=jsmith,ou=sales,o=rsa", the certificate DN in the browser is "0.9.2342.19200300.100.1.1=jsmith,ou=sales,o=rsa". In this case, enter "0.9.2342.19200300.100.1.1=jsmith,ou=sales,o=rsa" as the certificate DN value of the user in RSA Access Manager.

4. Restart your Entitlements and Authorization Servers after modifying the ldap.conf file.
5. Obtain the user's certificate DN and populate the DN field in the Administrative Console for each user. To do this, open each user's certificate (.cer or .crt file), copy the DN inside the certificate, and paste it into the DN field of the user entry in the Administrative Console. An example of a certificate DN is: cn=jsmith,ou=sales,o=rsasecurity,c=US.

See these parameters in cleartrust.properties: cleartrust.agent.auth.resource_list and cleartrust.agent.default_auth_mode. Also see the cleartrust.data.ldap.user.attributemap.certdn: dn parameter in ldap.conf.

Form-Based Authentication

By default, form-based authentication is enabled when you install the Agent. Several HTML and JSP forms are included to use with the authentication types. These forms are presented to users when they request a protected resource. You can customize the logon forms (used to collect the user's identity credentials) to your own organizational identity, and customize what your users see after logon by directing them to a specific web page. When form-based authentication is enabled and the user does not have adequate entitlements to access the requested protected page, the user is redirected to an access denied page.

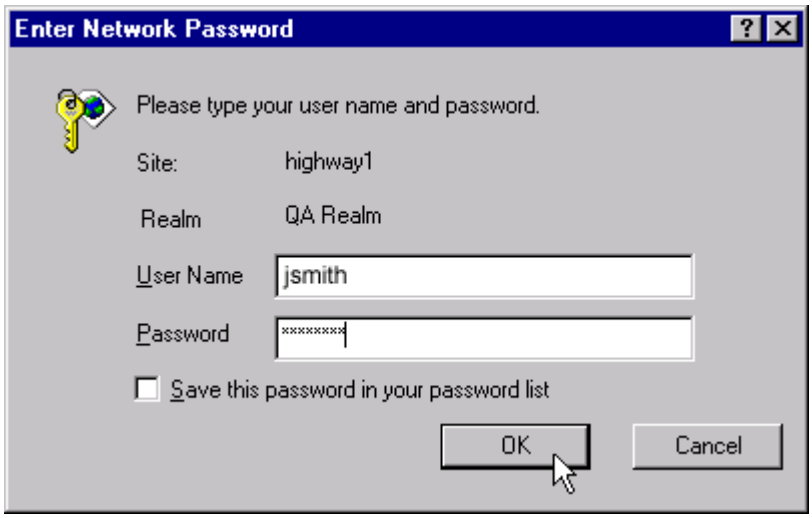
Authentication forms are set in cleartrust.properties, beginning with the cleartrust.agent.login_home_location. The HTML and JSP forms are installed as an application called Access Manager.ear in WebSphere. After authentication, the user can be directed to the original page or to a default home location. This is decided by the property cleartrust.agent.retain_url in cleartrust.properties.

For details on using multiple virtual hosts, see Chapter 17, “Troubleshooting.”

Form-based authentication is required to use single sign-on (SSO). This enables the Agent to maintain the user's credentials across requests.

Important: When designating directories to protect, use "/*" with caution. The form pages you specify in cleartrust.properties are explicitly unprotected by RSA Access Manager. Using "/*" to protect the entire application server web resources blocks access to any graphics or objects associated with those pages, causing them to display or function improperly.

If you disable form-based authentication, users see an HTTP authentication prompt native to the web browser. Disabling form-based authentication is recommended only if you are using Basic authentication.



When form-based authentication is disabled and the user does not have adequate entitlements to access the requested protected page, the user is given the standard "HTTP 404 - File Not Found" error message.

Pointing to Logon Forms

The format used to point to the appropriate logon form depends upon the type of web page being used.

For example:

- HTML pages

```
/ctappagent/ct_logon_<%language%>.html
```

HTML pages are the default, except when using RSA SecurID authentication.

```
/ctappagent/ct_logon.asp?CTAuthMode=BASIC&language=<%language%>
```

- JSP pages

```
/ctappagent/ct_logon.jsp?CTAuthMode=BASIC&language=<%language%>
```

Notice that the placement of the language tag is different for JSP pages than for HTML pages. Also, the CTAuthMode tag is used. CTAuthMode can be any of these authentication types: BASIC, NT, SECURID, CERTIFICATE, and CUSTOM.

JSP pages are required when using RSA SecurID authentication. The cleartrust.properties file is already properly configured.

Using Multiple Authentication Types

Granular resource-based authentication allows you to configure different authentication modes for different resources on any given RSA Access Manager protected application server. The Agent supports any or all of the authentication types listed for different resources on the application server. You can configure a global authentication type used for all resources, and you can at the same time specify different authentication types for selected resources at the URL level. For example, you could require users to present a digital certificate to access specific financial resources, while allowing users to access other applications with their Windows NT user name and password.

```
cleartrust.agent.default_auth_mode=BASIC
cleartrust.agent.auth_resource_list=/apps/*=NT,/financial/*=CER
TIFICATE
```

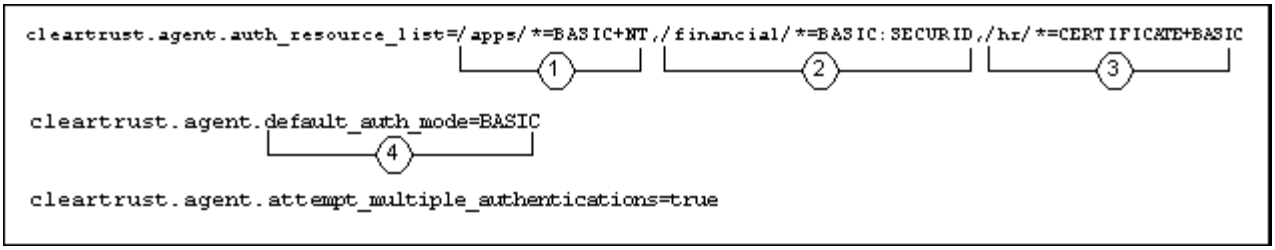
You can add as many specific URLs to the authentication resource list as you want, and each one can have a different authentication mode attached to it. The list is comma separated. If you do not specify a particular resource, the default authentication mode (.default_auth_mode) is used.

Note: If an invalid authentication type is specified in the .auth_resource_list parameter, the Agent starts up normally. However, when the user tries to access any resource in the web filter-protected application, the Agent throws a ServletError. Be sure to specify only valid authentication types in the .auth_resource_list parameter.

Multiple Authentication Chaining

It is possible to combine or chain together different authentication types for a particular resource. For example, the parameter settings in the following example dictate four different authentication requirements, each on different resources.

- Users trying to access anything in the /apps directory need to first provide Basic authentication, then Windows NT authentication.
- Users trying to access anything under /financial must supply either Basic or RSA SecurID authentication.
- Users trying to access anything in the /hr directory must have a digital certificate and provide Basic authentication.
- Any resources that are not explicitly mentioned in the parameter use the default, Basic authentication.



Rules

- cleartrust.agent.form_based_enabled must be set to true for multiple authentication.
- Boolean operators:
  - **AND.** The plus sign (+). Requires the user to fulfill both authentication types.
  - **OR.** The colon sign (:). Requires the user to fulfill any one of a set of authentication types in a sequential manner. The user is initially challenged to enter the first authentication type in the list.
  - The .attempt_multiple_authentications parameter tells the Agent to try the user's credentials against all authentication types listed without re-prompting the user if the first authentication attempt fails. This parameter applies only if using the OR operator.
  - The .attempt_multiple_authentications parameter tells the Agent to try the user's credentials against all authentication types listed without re-prompting the user if the first authentication attempt fails. This parameter applies only if using the OR operator.
  - For any given URL, you may string together multiple authentication types that are handled in either an AND or an OR relationship.
  - If you have configured password lockout, all of a user's logon failures, regardless of authentication type, are counted in the same counter. For example, a failure using Basic authentication followed by a failure using RSA SecurID authentication results in the counter being set to two. RSA Access Manager does not interact with any logon failure counters that are resident in your underlying user authentication types. For more information on password lockout and how it functions, see the Administrative Console Help or the RSA Access Manager Administrator's Guide.

Setting Up Authorization

In order to protect a web application using RSA Access Manager web filter and grant access rights to users, the web resource must be created in RSA Access Manager Servers, and the user or the user's group must be entitled to access the resource.

To set up authorization for web filter-protected web resources:

1. Log on to the RSA Access Manager Administrative Console.
2. Click **Define Resources > Servers > Add New.**

3. Enter the Server Name. Enter the name given in the cleartrust.agent.server_name parameter in the cleartrust.properties file.
4. Select "Enhanced App Server" for the server type.
5. Select the Product Name from the Product drop-down list. If the required Product Name is not listed in the Product drop down list, you can add it through the enhancedAppManufacturers parameter in the admingui.cfg file.
6. Enter the hostname of the WebSphere Application Server.
7. Click **Save.**
8. Click **Define Resources > Add Resources**.
9. Select the newly created "Enhanced App Server" as the Server Name from the drop-down list.
10. Click **Next.**
11. Select an application where you want to keep all your Application server-protected resources.
12. Click **Add Resource** and link the selected Application.
13. On the Add Resource page, specify the context root of the web application to be protected in the **Web Resource** field. For example, if the context root of the web application is "/stock", enter "/stock/*" in the **Web Resource** field.
14. Click **Save.**
15. Click **Done.**

Entitlements can now be granted for the users or groups who need to access the resource. For more details, see the *RSA Access Manager Servers* documentation.

Protecting J2EE Resources Using User Registry and Authorization Table Interface

Primary responsibility for creating security policies lies with the administrators that develop and deploy a WebSphere application. The protection of J2EE resources is determined by roles that are created as part of application development, and their mapping to the RSA Access Manager users and groups during and after the Agent deployment.

Basically, the developer defines what, and the deployment administrator defines who. The developer defines what resources must be protected and gives a name to the security roles protecting them. Then the deploying administrator defines who belongs to those roles by mapping them appropriately to actual users and groups.

After the Agent is installed, the assignment of roles to users or groups is managed through the RSA Access Manager Administrative Console.

Overview of Security Roles in WebSphere Application Development

When building an application, a developer indicates which WebSphere resources must be protected, and defines security roles controlling access to them. For instance, a banking application might require separate roles for managers and supervisors, in which managers are authorized to transfer money between accounts while supervisors are only allowed to view the accounts. In this case, the manager role is granted access to the J2EE resources that provide the transfer functionality, but the supervisor role is denied this access.

This assignment of roles is accomplished through the WebSphere Application Assembly Tool, or directly through manipulation of the application Deployment Descriptor. Details of these tools and procedures are provided in the *IBM WebSphere* documentation.

Creating WebSphere Security Roles in RSA Access Manager

WebSphere Application Server is based on the J2EE role-based security model. For RSA Access Manager to work with the WebSphere server's role-based security framework, the Agent is designed to map WebSphere roles to RSA Access Manager groups. These mappings are established either manually or using the migration tool (synctool) provided with the Agent.

Note: The RSA Access Manager Server already supports a concept called "Roles" for delegated administration. These "Roles" are entirely different from the security roles defined in WebSphere applications.

Mapping Conventions

The Agent uses the following naming convention to map a WebSphere security role to an RSA Access Manager group that represents the role, and vice versa:

- *RoleName_ApplicationName_ServerName_CTROLE*

where:

  - *RoleName* is the name of the security role that is defined in the application referenced by ApplicationName.
  - *ApplicationName* is the logical name of the deployed application.
  - *ServerName* is the unique name of the WebSphere server that is represented in RSA Access Manager. This value must be the same as the cleartrust.agent.server_name property value in the cleartrust.properties file.

All the WebSphere security roles in RSA Access Manager must end with a tag called CTROLE.

For example:

Consider a **Manager** role that is associated with an enterprise application called **BankingApp**. If the unique WebSphere server name is **WASServer**, then the RSA Access Manager group that the Agent refers to for this Manager role is **Manager_BankingApp_WASServer_CTROLE**.

Mapping Users or Groups to Security Roles (Authorization)

After creating the required WebSphere security roles as groups in RSA Access Manager, these security roles can be granted to any user or group in RSA Access Manager by adding the user or the group as a member of the newly mapped group.

For example:

To grant the WebSphere Manager role to a user called Tom, add **Tom** as a member user of the RSA Access Manager group **Manager_BankingApp_WASServer_CTROLE**, through the RSA Access Manager Administrative Console.

To grant the WebSphere **Manager** role to an RSA Access Manager group called BankManagers, add BankManagers as a member group of the group **Manager_BankingApp_WASServer_CTROLE**, through the RSA Access Manager Administrative Console. By doing this, all the users under the RSA Access Manager **BankManagers** group have the WebSphere Manager role.

While deploying the applications in WebSphere, it is possible to map roles to users or groups in RSA Access Manager. This is the traditional way to assign roles to users or groups in earlier RSA Access Manager Agents for WebSphere or in WebSphere itself. This is required because WebSphere does the role-based authorization by itself.

RSA Access Manager Agent 5.0 SP1 and later for WebSphere, the Agent can perform role-based authorization for J2EE resources. The role-to-user or role-to-group mapping that is done during WebSphere application deployment is not considered role-based authorization for RSA Access Manager. Only the roles that are mapped to user or groups in RSA Access Manager, as described in this section, are considered for role-based authorization. The Agent also has a tool called synctool that is capable of populating already deployed applications and their user or group role mappings to RSA Access Manager Servers. This tool can be used to migrate existing applications and their role mappings to RSA Access Manager Servers.

Running the Agent Synctool

In order for the existing applications with security roles to work properly, all the roles with their role-to-user or role-to-group mappings must be created in the RSA Access Manager Server. After installation, restart the WebSphere Application Server and run the synctool, which is located under the /tools/ folder. This creates all the necessary roles with their role-to-user or role-to-group mappings in RSA Access Manager for all the existing applications.

The tool is designed to bring the application server-specific policies into RSA Access Manager. It is strictly a one-way process. That is, changes made to users or groups in RSA Access Manager are not synced back to the application server. For more information on this utility, see “Synctool” on page 103.

Protecting Applications with the Trust Association Interceptor

Trust Association Interceptors (TAI) can be used to provide single sign-on with external web servers that are protected with an RSA Access Manager Web Server Agent. Once the user is authenticated with an RSA Access Manager-protected external web server, the user does not need to reauthenticate with WebSphere to access resources that are protected with J2EE roles.

TAI provides only authentication support. Users accessing J2EE role-based applications still must go through authorization checks by WebSphere. For details relating to role-based authorization, see "Creating WebSphere Security Roles in RSA Access Manager" on page 60.

Configuration

The Agent installer automatically installs the RSA Trust Association Interceptor during installation. No extra configuration is needed.

The functionality of the RSA Trust Association Interceptor is based upon the following properties present in the cleartrust.properties file.

- Validating Sessions:

```
cleartrust.agent.websphere.tai_validate_sessions=True
```

When this property is set to True, TAI validates the session token set by the front-end web server Agent against the RSA Access Manager Servers. The authenticated user name is also retrieved from the session token passed by the external web server.

Do not set this property to False unless you are sure that no one can directly access the protected resource in WebSphere, bypassing the external web server, or that the resource is protected in the external web server so that no one can access the resource through the external web server without authenticating.

- User Header Configuration:

```
cleartrust.agent.websphere.tai_user_header=ct_remote_user
```

This property specifies the HTTP header name in the request, which specifies the name of the authenticated user. This HTTP header must be set by the RSA Access Manager Web Server Agent present in the external web server. TAI uses this HTTP header's value as the authenticated user.

This property is used only if cleartrust.agent.websphere.tai_validate_sessions is set to False. If .tai_validate_sessions is set to False, .tai_user_header must be configured properly.

For information about the HTTP header name that will be used to represent the authenticated user name, see RSA Access Manager Web Server Agent documentation.

- Required Headers Configuration:

```
cleartrust.agent.websphere.tai_requiredheaders=ct_auth_type:2
```

This property requires comma-separated header value pairs in the form HeaderName[:HeaderValue]. When this property is set, TAI expects these headers with the corresponding values to be present in the HTTP request coming from the external web server. If any of the headers is not present in the HTTP request, or if any of the values does not match what is expected, TAI fails the request. If HeaderValue is not configured for a particular HeaderName, then TAI checks for the presence of HeaderName alone in the HTTP request.

- Disable Header Value Check:

```
cleartrust.agent.websphere.tai_check_requiredheaders_value=false
```

This property configures TAI to check for the existence of the HTTP HeaderNames only, which are present in the .tai_requiredheaders parameter. The HeaderValues that are specified are discarded.

- Authentication user credentials:

```
cleartrust.agent.websphere.tai_create_subject=true
```

This property specifies whether to create the Subject by the RSA Access Manager Agent and attach it to the TAIResult.

If the subject is True then the authenticated user credential is added to the TAIResult and stored in the WebSphere. Otherwise the subject is not added to the TAI Result.

Single Sign-On with RSA Access Manager Web Filter

Typically TAI is used to provide Single Sign-On (SSO) with RSA Access Manager-protected external web servers. WebSphere invokes TAI whenever a J2EE role-protected web resource is accessed by a user, whether the user accesses it through an external web server or directly through the web container.

Consider the following scenario:

- A user accesses an RSA Access Manager Web Filter protected web application after providing proper credentials.
- From the same session the user tries to accesses a J2EE role-protected web application.
- WebSphere invokes TAI to check whether the request is already authenticated.
- If .tai_validate_sessions is set to True, TAI searches for the RSA Access Manager session cookie and validates it. It also extracts the authenticated user name from the session cookie.

If .tai_validate_sessions is set to False, then TAI searches for .tai_user_header, which is not present because the request is not coming from the external web server. In this scenario, the user is prompted to reauthenticate.

Moreover, even if the cleartrust.agent.websphere.tai_validate_sessions is set to True, SSO can fail if the cleartrust.agent.websphere.tai_requiredheaders is set to check values for custom header values exported by the web filter.

In order to have SSO between a web filter-protected web application and then a J2EE role-protected web application, the following setup is necessary:

1. cleartrust.agent.websphere.tai_validate_sessions=True
2. cleartrust.agent.websphere.tai_requiredheaders=.tai_requiredheaders must be left empty

Using User Properties in Required Headers Field

RSA Access Manager Web Server Agents can be configured to publish user properties in the HTTP header after authentication. These user properties can be used in cleartrust.agent.tai_requiredheaders (parameter) property.

For more details about publishing user properties in the HTTP headers, see the cleartrust.agent.userprops parameter in the webagent.conf file of the web server Agents.

Previous versions of RSA Access Manager Agents cannot publish user properties. For details regarding publishing user properties, see the documentation of the RSA Access Manager Web Server Agent that is used in the external web server.

Publishing and Using User Properties in RSA Trust Association Interceptor (TAI)

You can configure RSA Access Manager to automatically publish user properties to the HTTP header upon successful authentication. This allows you to introduce into the session any user information that you have defined as custom user properties in the Administrative Console or Administrative API.

You can publish all available custom properties or only selected properties. Properties are published to the session after successful authentication.

Basic Configuration

By taking the appropriate steps in your administrative tools and Agent configuration file, you can publish all available custom properties or only selected properties. Note these requirements:

- **Marking properties for export with administrative tools.** Each property to be exported must be marked as Export/Publish when the RSA Access Manager administrator creates or modifies the property in the Administrative Console or through a custom Administrative API program. For more information, see the *Administrative Console Help* or the *RSA Access Manager Developer's Guide*.
- **Adding properties for export to the Agent configuration file.** Each property must be added to the cleartrust.agent.userproperties parameter in cleartrust.properties, unless the parameter is set to the wildcard "*", which publishes all properties marked for Export/Publish. The list is comma separated.

Protecting Web Services

Web services in simpler terms are web-based enterprise applications that use open, XML-based standards, and transport protocols to exchange data with calling clients. In a typical web services scenario, a business application uses the SOAP protocol over HTTP/HTTPS to send a request to a service at a URL. The service receives the request, processes it, and returns a response. An often-cited example of a web service is that of a stock quote service, in which the request asks for the current price of a specified stock and the response returns the stock price.

Web services can be protected using J2EE roles or RSA Access Manager web filters.

Protecting WebService with J2EE Roles

Since web services are tunneled over HTTP/HTTPS protocol, they take advantage of the security infrastructure provided by the web container. J2EE role-based security can be applied to the web application that hosts a web service. In such a case, authentication and authorization is handled by RSA Access Manager through the Custom User Registry and the Authorization Table Provider.

Basic and Certificate authentication types are supported when using role-based security.

- Setting Up Role-Based Authentication
- Setting Up Role-Based Authorization

Setting Up Role-Based Authentication

1. Add the required roles to the Deployment Descriptor during the web service development.
2. If the web service is an EJB, roles can be applied to the individual methods. Otherwise roles can be applied to the whole web application. If method-level protection is needed and the web service is not an EJB, you can create an enterprise bean with methods matching the web service operations. These EJB methods perform no operation; they are only dummy entities for applying security. Existing WebSphere authentication mechanisms can be applied to the enterprise bean. Before any web service operation is invoked, a call is made to the EJB method. If authorization is granted, the web service is invoked.

For more information about how to set up operation-level security for web services, see the IBM documentation.

Setting Up Role-Based Authorization

Once roles have been defined in the application, the roles must be assigned to users or groups for authorization. To assign the roles that are defined in the application, follow the steps in “Creating WebSphere Security Roles in RSA Access Manager” on page 60.

Protecting Web Services with RSA Access Manager Web Filter

Web services can also be protected using the RSA Access Manager web filters. The web filters used for protecting web services are different than those used to protect web applications. The web filter for web services is com.rsa.cleartrust.websphere.webservice.CTLoginFilter.

Basic, Certificate, and Windows NT authentication types can be used to protect web services when web filters are used.

Setting Up Web Filter-Based Authentication

To set up Web Filter-Based authentication:

1. Web services are invoked using a SOAP RPC router servlet that is mapped as a URL in the web application. To protect the web service, apply the web filter to the SOAP RPC router servlet. Add the following filter and filter-mapping tags to the deployment descriptor of the web application that hosts the web service.

```
<filter-name>CTLoginFilter</filter-name>
<filter-class>com.rsa.cleartrust.websphere.webservice.CTLogi
nFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>CTLoginFilter</filter-name>
<url-pattern>/servlet/*</url-pattern>
</filter-mapping>
```

2. Protect the SOAP RPC router in the RSA Access Manager Servers.

For example: If your context root of the application is “stockquote”, the URL is /stockquote/servlet/rpcrouter. Protect this URL in the RSA Access Manager Servers.

3. Specify the authentication type in cleartrust.properties. If the authentication type is not specified, the default authentication type configured in cleartrust.properties is used.

For example: cleartrust.agent.auth_resource_list=/stockquote/servlet/rpcrouter=NT

Setting Up Web Filter-Based Authorization

To set up Web Filter-Based authorization:

1. Log on to the RSA Access Manager Administrative Console.
2. Grant entitlements to the users or groups who are required to access the web service.

Protecting IBM Portal Server Resources

RSA Access Manager Agent for WebSphere provides support for Portal Server Authentication through Custom User Registry and Trust Association Interceptor. The Agent currently does not support granular Authorization of Portal resources. Authorization of Portal Resources such as portlets, pages, and so on must be managed through the Portal administration console.

IBM Portal Server Terminology

Portal Server. Portal Server is an instance of WebSphere Application Server that is responsible for rendering portal contents.

Trust Association Interceptor. Responsible for establishing trust between WebSphere Application Server and Authentication proxy.

User Registry. A database, which contains WebSphere's user and group information.

LDAP Registry. User Registry, which stores WebSphere's user and group information in a Directory Server.

Custom User Registry. User Registry, which stores WebSphere's user and group information in a custom data store.

Member Repository. A database where Portal Server stores Portal user accounts and other user-related information. Portal Server supports LDAP and Database as its Member Repository.

Portal Server can be configured to use its own native authentication mechanism or use authentication support provided by WebSphere Application Server such as User Registry and Trust Association Interceptor. The following table summarizes the configurations that are required for Portal Server to support WebSphere Application Server Security.

	Logon Type	Member Repository	User Registry
1	Direct Portal Logon	Database	Custom User Registry Provided by Portal Server
2	Direct Portal Logon	LDAP	LDAP Registry
3	Trust Association Interceptor	LDAP	LDAP Registry

In all the preceding configurations, Member Repository and User Registry use the same back-end data store. If Member Repository and User Registry do not share the same back-end data store, it is necessary to duplicate user entries in both of the back-end data stores. The Agent supports configurations 2 and 3. For configurations 2 and 3, the RSA Access Manager Server Data Store must be the same LDAP that is used by the Portal Server Member Repository.

However, RSA Access Manager Custom User Registry can be used for Portal Server Authentication by duplicating user entries and providing mapping between Portal Server users and RSA Access Manager users. If RSA Access Manager and the Member Repository share the same LDAP back-end data store, it is not necessary to duplicate user entries. Only the mapping of users needs to be configured. With these extra configurations, the Agent supports the following setup for Portal security.

	Logon Type	Member Repository	User Registry	RSA Access Manager Data Store	Mapping Required	Duplication of Users
1	DPL/TAI	LDAP	Custom User Registry Provided by RSA Access Manager	LDAP	Yes	No
2	DPL/TAI	LDAP	Custom User Registry Provided by RSA Access Manager	Database	Yes	Yes
3	DPL/TAI	Database	Custom User Registry Provided by RSA Access Manager	Any	Yes	Yes

- Direct Portal Logon (DPL)
- Trust Association Interceptor (TAI)

Configuring Portal Server for Using Websphere Security

When a Portal user logs in or out of the Portal, the Portal Server executes a certain login or logout command. The class names of these commands are specified in the property file PortalServer_Root/config/properties/ConfigService.properties.

Ensure that the login or logout and session validator commands are configured as follows:

```
command.login = LoginUserAuth
command.logout = LogoutUserAuth
command.sessionvalidator=SessionValidatorAuth
```

Mapping Users

The Member Repository stores the user entries in Distinguished Name (DN) format. For example, the user "john" is uniquely identified as cn=john,cn=users,dc=rsasecurity,dc=com in the Member Repository. If the RSA Access Manager Custom User Registry is used for Portal Authentication, then the user ID returned by the RSA Access Manager Custom User Registry is not be found in the Member Repository, as the RSA Access Manager user ID is not represented in DN format. To provide a mapping between the RSA Access Manager user ID and users in the Member Repository, configure the following properties in cleartrust.properties.

```
cleartrust.agent.portal.convert_uid_to_dn=true
cleartrust.agent.portal.user_dn_format=cn=<%uid%>,cn=users,dc=r
sasecurity,dc=com
```

RSA Custom User Registry converts the user ID to the format specified in the user_dn_format parameter by replacing the <%uid%> with the RSA Access Manager user ID.

If the Portal Server Member Repository is Active Directory, then the above mapping may not work, because Active Directory does not use the user ID to construct the user DN. In Active Directory, the user's full name is used to construct the user DN. For example, in Active Directory, the user ID "john" is represented as cn=john doe,cn=users,dc=rsasecurity,dc=com.

If RSA Access Manager users are created using the RSA Access Manager Administrative Console, this problem does not arise. If you create users without using the RSA Access Manager Administrative Console, you can perform the following steps to achieve successful mapping when using Active Directory as the Member Repository:

1. Copy rsactportal.jar from the AGENTBASE/lib folder to the PortalServer_Root/shared/ext folder.
2. In the PortalServer_Root/config/properties/LoaderService.properties file, modify the command.path property as shown here:

   ```
   command.path =
   com.ibm.wps.engine.commands,com.ibm.wps.dynamicui.commands,c
   om.rsa.cleartrust.websphere.portal
   ```
3. In the PortalServer_Root/config/properties/ConfigService.properties file, modify the command.login property as shown here:

   ```
   command.login = CTPumaServiceImpl
   ```
4. Restart the Portal Server.

Note: The Agent installer does not undo the configuration changes for Portal Server automatically during uninstallation. You must undo the configuration changes manually after you uninstall the Agent.

User Principal Name (UPN) Support

If RSA Access Manager is configured to use UPN as the userid, then the following configuration changes needs to be done in the RSA Access Manager Agent for WebSphere. Configure the following properties in cleartrust.properties and synctool.properties:

```
cleartrust.agent.websphere.subject_everyone=was_everyone@rsa.co
m
cleartrust.agent.websphere.subject_allauthenticated=was_allauth
enticated@rsa.com
```

8 RSA Access Manager Agent 5.0 SP1 Upgrade

This section describes how to upgrade the RSA Access Manager Agent 5.0 version to Agent 5.0 SP1 version on WebSphere Application Server.

Note: RSA Access Manager Agent 5.0 SP1 supports upgrade from RSA Access Manager Agent 5.0 version only. If you are using version prior to Agent 5.0, you must upgrade to Agent 5.0 before upgrading to Agent 5.0 SP1.

Upgrading to RSA Access Manager Agent 5.0 SP1

The upgrade of RSA Access Manager Agent 5.0 SP1 includes the following high-level tasks:

1. Backup the existing version of Agent 5.0. To backup the existing files, see Backup Existing Version.
2. For upgrade instructions see, Upgrade to RSA Access Manager Agent 5.0 SP1.
3. To verify the installation, start the WebSphere Application Server.
4. Deploy the Agent files, for instructions see Deploying Agent.

Backup Existing Version

1. Stop the WebSphere server on which Agent 5.0 is installed.
2. Navigate to the existing **WASBASE/lib/ext** directory and backup all the jar files.
3. Navigate to the existing **AGENT_HOME/tools** directory and backup the following files:
   - axm-was-agent-cacheflush-ejb-5.0.jar
   - ctencrypt.bat/sh
   - lockbox-tool.bat/sh
4. Navigate to the existing **AGENT_HOME/jars** directory and backup the files.

Note: When backing up files, do not rename it to old files, but remove it from the directory.

Upgrade to RSA Access Manager Agent 5.0 SP1

1. Navigate to **WASBASE/lib/ext** directory and copy the jar files from **/lib/** directory of Agent 5.0 SP1 package to **WASBASE/lib/ext** directory.
2. Navigate to **/tools/** directory in Agent 5.0 SP1 package and copy following files to **AGENT_HOME/tools** directory:

- axm-was-agent-cacheflush-ejb-5.1.jar
- ctencrypt.bat/sh
- lockbox-tool.bat/sh

3. Edit the tools to add **WAS_HOME** and **AGENT_HOME** variables**.**
4. Navigate to **WAS_PROFILE_HOME/properties** directory and update the cleartrust.properties file with the additional properties provided in **/conf/config_parameter.txt** file of the Agent 5.0 SP1 package.
5. Copy and replace the **debugenabled.properties** and **debugdisabled.properties** files present in the /conf/ directory in the Agent 5.0 SP1 to **WAS_PROFILE_HOME/properties** directory.
6. Start WebSphere Application Server.

Note: Post installation, to run cacheflush utility, it is recommended to add the resources under the cacheflush application in Access Mananger Admin console, as Entitlements to either was_allauthenticated or was_everyone users.

Deploying Agent

1. Log into the WebSphere Application Server administrative console.
2. Click Deployments, under Domain Structure Panel, and uninstall the following applications:
   - axm-was-agent-cacheflush-ear-5.0.ear
   - axm-was-agent-filterui-ear-5.0.ear
3. Click Deployments, under Domain Structure Panel, and install the following applications present in **/webapps/** directory in Agent 5.0 SP1 package:
   - axm-was-agent-filterui-ear-5.1.ear
   - axm-was-agent-cacheflush-ear-5.1.ear

Note: You must make sure that both axm-was-agent-filterui-ear-5.1.ear and axm-was-agent-cacheflush-ear-5.1.ear applications are active and started.

Optional Tasks

1. Navigate to the **/webapps/** directory in Agent 5.0 SP1 package and copy the available resources to the existing **AGENT_HOME/webapps** directory.
2. Navigate to the **/jars** directory in Agent 5.0 SP1 package and copy the available resources to the existing **AGENT_HOME/jars** directory.

9 Configuring Proxy Environments

When using the Agent to protect application servers that reside behind a proxy server, you must consider some additional configuration parameters. The following settings are applicable only to the web resources residing in the application server that are protected by web filters.

IP Checking. Specifies whether to validate the IP address stored in the cookie to determine if it is coming from a trusted source. For users accessing protected web resources, through a proxy server, the IP address must be verified only once (by the Agent installed on the proxy server).

Returning Cookies to the Client. If you are using a proxy/web server to front several application servers, potentially two SSO cookies can be returned to the client machine: the cookie generated by the proxy/web server Agent and the cookie generated by the application server Agent. In proxy environments, configure the Agents on your application servers to set the cookie IP address to remain that of the client instead of the proxy server. For more information, see "Returning Cookies to the Client" on page 75.

IP Checking

The IP Checking feature includes:

Excluding Certain Agents From the IP Check

The IP check feature validates the IP address stored in the cookie to determine if it is coming from a trusted source. The Agent compares the IP address of the client host with the IP address stored in the cookie before it uses the cookie for SSO.

In a typical proxied implementation, the proxy server and all content servers are protected by RSA Access Manager Agents. Enable IP checking only on the proxy server. If IP checking is enabled on the content servers, when the content server Agent checks the originating IP address of an incoming request, it sees only the IP address of the proxy server or the firewall (if the content servers are behind a firewall), causing the IP check to fail.

By default, IP checking is enabled when the Agent is installed. See .cookie_ip_check in cleartrust.properties.

Excluding Proxy Servers and Firewalls from the IP Check

Frequently, your proxy servers send requests to the RSA Access Manager Authorization Servers. You may not want IP checking for these requests because they originate from a trusted host inside your network. To exclude a proxy server or firewall, list the IP address of the server in the .ip_check_exclusion_list parameter in cleartrust.properties on the application server. Separate the IP addresses of multiple servers with a comma. For example:

```
cleartrust.agent.ip_check_exclusion_list=111.222.33.44,111.22
2.33.45
```

Any requests that do not arrive by way of the proxy server or firewall are still checked for a valid originating IP address when .cookie_ip_check is enabled. This parameter is not read or used by the system if the .cookie_ip_check parameter is disabled.

Excluding All Agents Within the Same Class C Subnet From IP Check

In a 32-bit IP address, the number of bits used to identify the network and the host vary according to the network class of the address. In a Class C network, the first 3 bits, or the high-order bits, are always 110. The next 21 bits are used to define the Class C network, and the final 8 bits are used to identify the host.

Instead of listing all proxy servers and firewalls in the IP check exclusion list, you can use a more general setting that instructs the Agent to forgo IP checking for any request coming from an IP address on the same Class C subnet (or internal network) as the Agent. To do this, enable .allow_subnet_masking in cleartrust.properties. This parameter is not read or used by the system if the .cookie_ip_check parameter is disabled.

Returning Cookies to the Client

In a typical proxied setting, the Agent is installed on the proxy and content servers. As a result, when a user requests a protected resource, both the Agent on the proxy server and the Agent on the content server attempt to generate and return an SSO cookie. To avoid this redundancy, configure the content server Agents to suppress generation of the cookie. This ensures that the browser receives only the cookie generated by the proxy server Agent, preserving the authentication relationship between the browser and the proxy server Agent.

To suppress generation of the SSO cookie, add the IP addresses of all proxy servers and firewalls as a comma-separated list in the .cookie_exclusion_list parameter in the Agent configuration file of the content server. For example:

```
cleartrust.agent.cookie_exclusion_list=111.222.33.44,111.222.33
.45
```

10 Publishing User Properties

You can configure RSA Access Manager to automatically publish user properties to the HTTP header upon successful authentication. This allows you to introduce into the session any user information that you have defined as custom user properties in the RSA Access Manager Administrative Console or Administrative API.

You can publish all available custom properties or only selected properties. Properties are published to the session after successful authentication.

Basic Configuration

By taking the appropriate steps in your administrative tools and Agent configuration file, you can publish all available custom properties or only selected properties. Note these requirements:

- **Marking properties for export with administrative tools.** Each property to be exported must be marked as Export/Publish when the RSA Access Manager administrator creates or modifies the property in the Administrative Console or through a custom Administrative API program. For details, see the Administrative Console Help or the *RSA Access Manager Developer's Guide.*
- **Adding properties for export to the Agent configuration file.** Each property must be added to the cleartrust.agent.userproperties parameter in cleartrust.properties, unless the parameter is set to the wildcard "*", which publishes all properties marked for Export/Publish. The list is comma separated.

Handling Backward Cookie Compatibility

If you have Web Servers still running Agent 1.0 released with RSA ClearTrust 4.6.1, set this parameter in aserver.conf:

```
cleartrust.aserver.token_version=0
```

and this parameter in cleartrust.properties:

```
cleartrust.agent.cookie_name=ctrust-session-v002d
```

This tells RSA Access Manager Agent 5.0 SP1 to issue cookies in a format compatible with Agent 1.0. This configuration is not required to communicate with newer versions, which share a common format with RSA Access Manager Agent 5.0 SP1. If you are using RSA SecurID as your authentication type and have a mixed environment of Agents 3.0, 4.7, 4.7 SP1 along with RSA Access Manager Agent 5.0 SP1, you cannot use RSA SecurID system-generated PINs. User-defined PINs are supported.

Implementing URL Retention

The following settings are applicable only to the web resources residing in application servers that are protected by web filters.

When used with form-based authentication, URL retention enables the web server to retain original page requests instead of sending users to a default location (such as the home page) after they log in. URL retention involves redirecting and storing the original page request in a cookie or in a query string, as described in this outline of the process:

- When requesting a protected resource, the user is redirected to the appropriate logon page as specified in the .login_form_location_ parameter in cleartrust.properties.
- The original URL of the requested resource is stored in one of two ways: either in the encrypted session cookie, or in the login form query string as ct_orig_uri. This is specified in the parameter .retain_url.use_query_string.
- Once the user is authenticated, the user is redirected back to the originally requested protected resource.

Configuring URL Retention

To configure URL retention, changes are required in the cleartrust.properties file. The setup steps differ between cookie-based and query string-based URL retention. By default, cleartrust.properties is configured with cookie-based URL retention parameters.

To configure cookie-based URL retention:

1. Verify that these parameters in cleartrust.properties are enabled with the values shown:

   ```
   cleartrust.agent.form_based_enabled=True
   cleartrust.agent.retain_url=True
   cleartrust.agent.retain_url.use_query_string=False
   ```

2. Specify the location of the default home page. Cookie-based URL retention uses the HTML form provided in the RSA Access Manager logon application.

   ```
   cleartrust.agent.login_home_location=/ctappagent/ct_home_<%l
   anguage%>.html
   where <%language%> is optional for this parameter.
   ```

3. Specify the location of the default logout page to display when a user ends an RSA Access Manager session.

```
cleartrust.agent.logout_form_location=/ctappagent/ct_logout_
<%language%>.html
where <%language%> is optional for this parameter.
```

4. Specify the location of the logon pages to display to your users.

- **Basic** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_basic=/ctappagent/ct_lo
gon_
        <%language%>.html
cleartrust.agent.login_error_user_location_basic=/ctappagent
/ct_logon_failed_
        <%language%>.html
cleartrust.agent.login_error_pw_location_basic/ctappagent/ct
_logon_failed_
        <%language%>.html
```

- **Windows NT** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_nt/ctappagent/ct_logon_
nt_<%language%>.html
cleartrust.agent.login_error_user_location_nt=/ctappagent/ct
_logon_failed_nt_
        <%language%>.html
cleartrust.agent.login_error_pw_location_nt=/ctappagent/ct_l
ogon_failed_nt_
        <%language%>.html
```

- **RSA SecurID** - Set the following parameters and associated logon forms. RSA SecurID always uses JSP versions of the logon pages.

```
cleartrust.agent.login_form_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&language=<%langu
age%>
cleartrust.agent.login_form_location_sid_nexttoken=

/ctappagent/ct_securid.jsp?SecurIdMode=nexttoken&language=<%
language%>
cleartrust.agent.login_form_location_sid_passcode=

/ctappagent/ct_securid.jsp?SecurIdMode=passcode&language=<%l
anguage%>
cleartrust.agent.login_form_location_sid_newpin_user_selecta
ble=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=use
rselectable
cleartrust.agent.login_form_location_sid_newpin_sys_generate
d=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=can
notchoose
```

```
cleartrust.agent.login_form_location_sid_newpin_user_specifi
ed=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=mus
tchoose
cleartrust.agent.login_form_location_sid_newpin_display=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=dis
play
cleartrust.agent.login_error_sid_newpin=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=
        mustchoose&CTLoginErrorMsg=Invalid%20new%20pin
cleartrust.agent.login_error_user_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg=
        Login%20Unsuccessful
cleartrust.agent.login_error_pw_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg=
        Login%20Unsuccessful
cleartrust.agent.login_error_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg=
        Login%20Unsuccessful

SecurIdMode must be specified in the query string of the URL
with one of the following values: nexttoken, newpin, or
passcode. If SecurIdMode is not set, the default value is
nexttoken.
```

- **Custom** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_custom=/ctappagent/ct_l
ogon_custom_
        <%language%>.html
cleartrust.agent.login_error_user_location_custom=
        /ctappagent/ct_logon_failed_custom_
        <%language%>.html
cleartrust.agent.login_error_pw_location_custom=
        /ctappagent/ct_logon_failed_custom_
        <%language%>.html
```

Note: Do not change the ct_logon.xxx page name. You can modify the contents of the file as long as you maintain ct_logon as the filename. Use %20 to define spaces in the previous arguments.

To configure query string-based URL retention:

1. Verify that these parameters are enabled with a value of "True" in cleartrust.properties, as shown:

```
cleartrust.agent.form_based_enabled=True
cleartrust.agent.retain_url=True
```

2. Change the value of .retain_url.use_query_string to "True", as shown:

```
cleartrust.agent.retain_url.use_query_string=True
```

3. Specify the location of the default home page.

```
cleartrust.agent.login_home_location=/ctappagent/ct_home.jsp?la
nguage=<%language%>
   where ?language=<%language%> is optional for this parameter.
```

4. Specify the location of the default logout page to display when a user ends an RSA Access Manager session.

```
cleartrust.agent.logout_form_location=/ctappagent/ct_logout.
jsp?language=<%language%>
where ?language=<%language%> is optional for this parameter.
```

5. Specify the location of the logon pages to display to your users.

- **Basic** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_basic=

/ctappagent/ct_logon.jsp?CTAuthMode=BASIC&language=<%languag
e%>
cleartrust.agent.login_error_user_location_basic=

/ctappagent/ct_logon.jsp?CTAuthMode=BASIC&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
cleartrust.agent.login_error_pw_location_basic=

/ctappagent/ct_logon.jsp?CTAuthMode=BASIC&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
```

- **Windows NT** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_nt=
        /ctappagent/ct_logon.jsp?CTAuthMode=NT&
        language=<%language%>
cleartrust.agent.login_error_user_location_nt=

/ctappagent/ct_logon.jsp?CTAuthMode=NT&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
cleartrust.agent.login_error_pw_location_nt=

/ctappagent/ct_logon.jsp?CTAuthMode=NT&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
```

- **RSA SecurID** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&language=<%langu
age%>
cleartrust.agent.login_form_location_sid_nexttoken=

/ctappagent/ct_securid.jsp?SecurIdMode=nexttoken&language=<%
language%>
cleartrust.agent.login_form_location_sid_passcode=
```

```
/ctappagent/ct_securid.jsp?SecurIdMode=passcode&language=<%l
anguage%>
cleartrust.agent.login_form_location_sid_newpin_user_selecta
ble=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=use
rselectable
cleartrust.agent.login_form_location_sid_newpin_sys_generate
d=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=can
notchoose
cleartrust.agent.login_form_location_sid_newpin_user_specifi
ed=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=mus
tchoose
cleartrust.agent.login_form_location_sid_newpin_display=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=dis
play
cleartrust.agent.login_error_sid_newpin=

/ctappagent/ct_securid.jsp?SecurIdMode=newpin&NewPinMode=
        mustchoose&CTLoginErrorMsg=Invalid%20new%20pin
cleartrust.agent.login_error_user_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg
        =Login%20Unsuccessful
cleartrust.agent.login_error_pw_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg=
        Login%20Unsuccessful
cleartrust.agent.login_error_location_securid=

/ctappagent/ct_logon.jsp?CTAuthMode=SECURID&CTLoginErrorMsg=
        Login%20Unsuccessful
SecurIdMode must be specified in the query string of the URL
with one of the following values: nexttoken, newpin, or
passcode. If SecurIdMode is not set, the default value is
nexttoken.
```

- **Custom** - Set the following parameters and associated logon forms:

```
cleartrust.agent.login_form_location_custom=

/ctappagent/ct_logon.jsp?CTAuthMode=CUSTOM&language=<%langua
ge%>
cleartrust.agent.login_error_user_location_custom=

/ctappagent/ct_logon.jsp?CTAuthMode=CUSTOM&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
```

```
cleartrust.agent.login_error_pw_location_custom=

/ctappagent/ct_logon.jsp?CTAuthMode=CUSTOM&CTErrorLoginMsg=
        Login%20Failed&language=<%language%>
```

CTLoginErrorMsg can be any error message that you want.

Note: Do not change the ct_logon.xxx page name. You can modify the contents of the file as long as you maintain ct_logon as the filename. Use %20 to define spaces in the previous arguments.

Centralized Login

If you choose to centralize logon operations in your environment, you may need to adjust the relevant cleartrust.properties parameters, as described in this section.

Note: Centralized logon is not supported with the Certificate authentication type or non-form-based authentication.

This Agent supports centralized logon with query-string based URL retention only. Using cookie-based URL retention with centralized logon is not supported.

To configure centralized logon with query string-based URL retention:

1. Adjust the logon form location parameters to point to the appropriate host using the full URL, not a relative URL. In this example for Basic authentication, "host-b.domain.com" is the host targeted to perform logon operations:

   ```
   cleartrust.agent.login_form_location_basic=http://host-b.dom
   ain.com/cleartrust/ct_logon.jsp?CTAuthMode=BASIC
   ```

2. Verify that .form_based_enabled is set to "True".
3. Verify that .retain_url.use_query_string is set to "True".
4. Verify that .retain_url.use_full_url is set to "True". This is the default value.
5. Verify that .login_home_location, .login_form_location_*, .login_error_user_location_*, and .login_error_pw_location_* are set to jsp|asp versions of the forms in the host that provides centralized logon support.

11 General Configurations

RSA Access Manager Agent 5.0 SP1 supports logging of Agent logs to a Centralized log server. This feature facilitates distributed Agent deployments across the enterprise to redirect all the Agent logs to a central log server for better troubleshooting, maintenance and archiving.

RSA Access Manager Agent 5.0 SP1 also provides an option to limit the number of log files in the log directory.

This chapter details the Default and Centralized logging modes.

Logging Modes

RSA Access Manager Agent 5.0 SP1 supports the following logging modes:

- Default Logging
- Centralized logging

Default Logging

RSA Access Manager Agent 5.0 SP1 uses Log 4j framework for default logging.

In this mode **cleartrust.agent.centralized.logging.enable** parameter is disabled.

Limit the number of files in log directory

Agent 5.0 SP1 uses **org.apache.log4j.RollingFileAppender** appender to limit the numer of log files.

You can append maximum file size limit to file in the log directory by configuring the following parameter:

log4j.appender.cleartrustfile.MaxFileSize

You can also include the maximum number of backup files index value in the log directory by configuring the following parameter:

log4j.appender.cleartrustfile.MaxBackupIndex

Enabling Default Logging

This section expalins how the default logging can be configured using the debugenabled.properties file.

To enable default logging, set the following parameter in cleratrust.properties file present in **WAS_PROFILE_HOME/properties** directory to the following:

```
cleartrust.agent.debug_configuration=debugenabled.properties
```

Disabling Default Logging

This section expalins how the default logging can be removed using the debugdisabled.properties file.

To disable default logging, set the following parameter in cleratrust.properties file present in **WAS_PROFILE_HOME/properties** directory to the following:

```
cleartrust.agent.debug_configuration=debugdisabled.properties
```

Additional Information

The following section provides an overview of debugenabled.properties.

Following is the content with some values assigned as an example:

```
log4j.rootCategory=DEBUG, cleartrustfile, stdout
log4j.appender.cleartrustfile=org.apache.log4j.RollingFileAppen
der
log4j.appender.cleartrustfile.File=cleartrust.log
log4j.appender.cleartrustfile.Append=false
log4j.appender.cleartrustfile.MaxFileSize=5 MB
log4j.appender.cleartrustfile.MaxBackupIndex=2
log4j.appender.cleartrustfile.layout=com.rsa.cleartrust.common.
log.log4j.AgentLogPatternLayout
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=com.rsa.cleartrust.common.log.log4
j.AgentLogPatternLayout
log4j.appender.stdout.Target=System.out
```

In the above example, you can apprehend the following:

- The level of the root logger is defined as DEBUG and attaches appender named cleartrustfile and stdout to it.
- The appender cleartrustfile is defined as org.apache.log4j.RollingFileAppender and writes to a file named " cleartrust.log ".
- The maximum permissible size of each log file is 5MB.
- The maximum number that the output is allowed to reach before being rolled over to backup files.

When debug option is disabled, no information is logged, but you can configure to log at ERROR log level by uncommenting the following lines:

#log4j.logger.com.rsa=OFF

#log4j.logger.sirrus=OFF

Centralized Logging

Agent uses Log 4j framework to support centralized logging for logging messages. Log 4j framework consists of Appenders and loggers, that helps log messages on the basis of message type and format.

This mode includes configuring Agent for centralized logging with Access Manager Servers that are running on Java Virtual machine, with the configuration cleartrust.agent.centralized.logging.enable=True

Failover Logging

Agent also provides failover support, which is if the primary Access Manager Log server fails then the log files are logged to secondary log server, when single or multiple instances of log server (s) are configured.

When all or single host server(s) fail, then the failover support provided is similar to default logging.

Following are the two failover logging types supported:

- Failover logging using Single log server host- This includes failover logging support only when single instance Access Manager Log server host is configured. This failover mechanism is similar to default logging where in the default file **cleartrust.log** is used to log messages.
- Failover logging using multiple log server hosts-This includes failover logging support when multiple instances Access Manager Log server hosts are configured. When all the log server hosts fail, then the default file **cleartrust.log** is used to log messages.

Centralized Logging Configuration Parameters

Following parameters must be set, to utilize Centralized logging option:

Parameter	Description
cleartrust.agent.centralized.logging.enable	Specifies whether Centralized Logging is enabled or disabled in Application agents.
cleartrust.agent.logserver.clientId	Specifies the Agent instance that is logging to Centralized log server. The value of this parameter is added to the beginning of each log message. When several different agents are sending log messages to a Access Manager log server, this parameter can be used to identify the origin of each message. A value is appended to this parameter, which is displayed in the beginning of each log message.

cleartrust.agent.logserver_list	Specifies the Access Manager Log Server connection details. You can use this parameter to configure the Access Manager Log Server host and port details. Default value:None Allowed Value:host1:port1, host2:port2 **Note:** To use this parameter, you must set the following parameters: cleartrust.agent.centralized.logging.enable and cleartrust.agent.logserver.clientId
cleartrust.agent.logserver.log_delimiter	Specifies the separated entries in log messages in Access Manager Log server. Allowed Values: Any string or special characters.
cleartrust.aserver.log.server_reconnect_delay	When the Access Manager Log server connection is lost, this parameter specifies the number of milli seconds to wait before reestablishing the connection. Default Value:30000 ms (30 seconds) Allowed Value: A positive integer. **Note:** This parameter is only used for remote logging.

Agent uses AxM Socket Appender to establish connection with Access Manager log server using the following modes:

- Anon
- Auth

Important: RSA recommends the communication between Agent and Access Manager Log server using Auth mode.

Configuring Centralized Logging for RSA Access Manager Agent 5.0 SP1

Pre-requisites

- RSA Access Manager Servers along with Access Manager Log Server must be installed and running.
- Required configuration parameters are set in cleartrust.properties file.

To configure Centralized Logging in the application agent, complete the following steps:

1. Set the connection mode to Auth or Anon, for connecting to Access Manager Log server.
2. Specify the following parameter in cleartrust.properties file:

- cleartrust.agent.centralized.logging.enable
- cleartrust.agent.logserver.clientId
- cleartrust.agent.logserver_list
- cleartrust.agent.logserver.log_delimiter
- cleartrust.aserver.log.server_reconnect_delay

Note: If clientID is not configured, IP Address of the Agent host is appended. If connection is not established to the configured Access Manager logserver host, log events are written by default to the Agent Host log file by name cleartrust.log file.

12 Configuring SSO

The following settings are applicable only to the web resources residing in application servers that are protected by web filters. SSO can be configured between RSA Access Manager-protected application servers and also between RSA Access Manager protected application servers and web servers. Perform these steps on every application and web server that will be participating in SSO.

To configure SSO:

1. Install the RSA Access Manager Agent on each application server and web server you want to protect (if you have not already done so).
2. Set these parameters in each RSA Access Manager-protected application and web server Agent's configuration file (cleartrust.properties for application server Agents and webagent.conf for web server Agents):

- Enable single sign-on:

  ```
  cleartrust.agent.sso=True
  ```

- Set the domain name. The domain name that is set in the SSO cookie at runtime must be the same for all applications and web servers participating in SSO:

  ```
  cleartrust.agent.cookie_domain=.company.com
  Domain name values need a period before the domain name. If this parameter is left unset, the browser uses the name of the host server as the domain name in the SSO cookie. This allows SSO to work for the specific server only.
  ```

3. Review the other SSO and cookie Agent parameters to determine if you need to adjust the default settings for your application server environment. See .session_lifetime, .idle_timeout, .secure, .fudge_factor, .cookie_touch_window, and .auto_challenge parameters in the cleartrust.properties.

13 Clustered Environment Agent

This Agent supports WebSphere Network Deployment edition. In this scenario, the Agent is installed in the Deployment Manager node. The Agent installer copies all the Agent's libraries and configuration files to all the nodes that are managed under the Deployment Manager. The Agent installer also installs the RSA Access Manager Agent login application and cacheflush tool on all the servers and clusters present in the Deployment Manager.

The Agent considers all the clusters and individual servers that come under the Deployment Manager to be a single server. The Application Server Resources that are managed under the Deployment Manager must be kept under a single server in RSA Access Manager.

Whenever a parameter is modified in the cleartrust.properties file after installation, the properties file must be manually synchronized across all the nodes managed under the Deployment Manager.

Adding a Node in the Agent-Protected Deployment Manager

When a new node is added to the Deployment Manager on which the Agent is already installed, you must manually copy the Agent libraries and the cleartrust.properties file to the newly added node.

To add a new node to the Agent deployment:

1. Copy the Agent libraries from the *DEPLOYMENT MANAGERHOST/lib/ext* folder to the /lib/ext folder of the new node.
2. Copy the cleartrust.properties, debugenabled.properties, debugdisabled.properties, and log4j.properties file from the *DEPLOYMENT MANAGERHOST/properties* folder to the /properties folder of the new node.
3. If the Agent uses certificates, copy all of the certificates to the new node. Also update the cleartrust.properties file copied in step 2 with the certificate path of the new node. Certificates can be kept under the properties folder or the etc folder of the new node. While updating the certificate path, provide the full path of the certificate file. The Agent does not accept relative paths.
4. Add the new node. Since Global Security is enabled in the Deployment Manager, use the -username and -password options while running the “addnode” script.
5. To create a lockbox if node is on a different machine, perform the following:
   a. Change directories to **WASPROFILE/bin**, and open **setupCmdLine.sh** file.
   b. Add the following parameter to the setupCmdLine.sh file:
   export LD_LIBRARY_PATH=AGENTBASE/cst:$LD_LIBRARY_PATH

For more information on creating lockbox, see Lockbox File Tool.

6. Start the node Agent of the newly added node.

An alternative approach is to reinstall the Agent after adding the new node. Follow the steps given below to add a node:

1. Turn off Global Security through the Deployment Manager administrative console.
2. Restart the Deployment Manager and all the node Agents.
3. Add the new node to the Deployment Manager.
4. Reinstall the RSA Access Manager Agent.
5. Restart the Deployment Manager and all the node Agents.

Note: When a new node is added to the Deployment Manager after the Agent installation, the new node will fail to start. You must perform the preceding steps to add the node to the Agent deployment..

Removing a Node in the Agent-Protected Deployment Manager

To remove a node from the Deployment Manager, you must manually remove the Agent libraries and the cleartrust.properties, debugenabled.properties, debugdisabled.properties, and log4j.properties file from the corresponding node.

14 Internationalization Support

Internationalization support is provided for Access Manager application agent login pages version 5.0.

The internationalization feature is enabled for JSP login pages.

Setting up Internationalization

This section provides an overview to set up Internationalization for web agent logon pages.

No.	Task Description	Reference
1	**Language Selection.** You must select the language for the logon pages. The default language is English.	For more information see, Language Selection..
2	**Append Language Query String.** You must append the language query string in all the logon pages location. You can refer to the respective logon pages for configuration.	For more information see, Appending Language Query String..
3	**Property File.** You can also configure the default property file.	For more information see, Property File.
4	**Locale Specific Property File.** You must create the locale specific property file for the selected language. By default, a property file for English is provided.	For more information on JSP pages, Setting up Locale value and Locale specific File for JSP.
5	**Enable i18n Support.**	For more information see, Enable i18n support for JSP Login Pages.

Language Selection

You can define the supported languages by configuring the **cleartrust.agent.accepted_languages_list** parameter present in the cleartrust.properties file.
For example, **cleartrust.agent.accepted_languages_list =ja,fr-FR,en**.

For every HTTP request, the **Accept-Language** header is read to identify the client browser accepted Languages list.

The list is compared with the values specified for the parameter **cleartrust.agent.accepted_languages_list**:

- If there is a match, the matched language is selected as the language used for the request.
- If there is no match or the client browser does not send an 'Accept-Language' header, then language defined in the **cleartrust.agent.default_language**, is used.

By default **cleartrust.agent.default_language parameter** is set to **en** in the **cleartrust.properties** file. You can set the required language code supported as the default language and create the corresponding language property file.

Appending Language Query String

Append **language=<%language%>** as a query string in all the logon pages locations in the **cleartrust.properties** file. For example:

```
cleartrust.agent.login_form_location_basic=/cleartrust/ct_lo
gon.jsp?language=<%language%>
```

Property File

A property file is a resource bundle that stores locale-specific messages, labels and image paths. The property file name is saved using a standard naming convention that includes the basename combined with a locale name. The property file contains key variables to store the values of the messages, labels and image paths.

The location of the property file for JSP is mentioned below:

- JSP-**\WEB-INF\classes\com\rsa\cleartrust\i18n** of the deployed cleartrust web application folder.

The default property file is **LogonPages.properties** for JSP, which corresponds to the default language, English. If the selected locale specific property file is not present, then the values are read from the properties file corresponding to the default language defined.

You can perform the following with the property file:

- **Add new values.** To add new messages, error messages, labels or image paths, you must define a key and assign a value for it in the property file. Following this, you must also include the key in the logon pages.
- **Modify existing values.** You can modify the existing values for the keys defined in the property file.

Note: Do not delete or replace the default property file.

Setting up Locale value and Locale specific File for JSP

You must make a copy of the existing **LogonPages.properties** file, rename the file by appending the language code to the filename, and provide the values for the keys in the file specific to the language selected. For example, to set up the Locale Specific Property File for French, make a copy of the **LogonPages.properties** property file and rename it to **LogonPages_fr.properties**. Then provide the new values for the keys specific to the language. You can create similar property files for each of the languages supported.

Note: You must not modify or delete existing keys in the property file. The values entered for the keys in the language specific file must be in Unicode-UTF 8 format. To create Locale Specific Property file, use the editor that supports UTF-8 encoding. This is applicable to all JSP and Servlets property files.

Note: The property file needs to be converted from native-encoded characters to Unicode-encoded characters. You can use tools such as native2ascii.exe to perform the conversion.

Internationalization Support for Images

Place locale specific images in the **images** folder of the deployed **axm-jboss-agent-filterui-5.0.war** application for internationalization support for images.

Enable i18n support for JSP Login Pages

You can enable internationalization support after configuring the language parameters.

To enable internationalization support for JSP logon pages, complete the following steps:

1. Configure the accepted languages parameter and default language parameter in the **cleartrust.properties** file, that is **cleartrust.agent.accepted_languages_list** and **cleartrust.agent.default_language**.
2. Append **language=<%language%>** as a query string in all the login form locations in the **cleartrust.properties** file.

   For example:

   ```
   cleartrust.agent.login_form_location_basic=/cleartrust/ct_lo
   gon.jsp?language=<%language%>
   ```

3. Create a new locale specific property file, namely LogonPages_<locale>.properties file in **\WEB-INF\classes\com\rsa\cleartrust\i18n** folder where axm-jboss-agent-filterui-5.0.war file is deployed.

   where,

   the value of locale can be **< 2 letter ISO language code in lower case > or <2 letter ISO language code in lower case_2 letter ISO country code in UPPER case>**, and must correspond to the values specified in the **cleartrust.agent.accepted_languages_list** parameter of the **cleartrust.properties** file.

   - Country code is optional and can be specified when specific dialects are required. For example, if you are configuring for French language, without country code specific then **cleartrust.agent.accepted_languages_list = fr** must be configured in the cleartrust.properties file and a new **LogonPages_fr.properties** file must be created as locale specific property file.
   - If you are configuring for French language in France, then **cleartrust.agent.accepted_languages_list = fr-FR** must be defined in the cleartrust.properties file and a new **LogonPages_fr_FR.properties** file must be created as locale specific property file.

   Note: The default LogonPages.properties file that supports English is present in **\WEB-INF\classes\com\rsa\cleartrust\i18n** folder of the web application.

4. In the locale specific property file, update the value for all messages, labels and error messages using unicodes in **UTF-8 encoding**.

   For example, in the default property file, a property named UserID, is defined as UserID=User ID. You must add the values in the property file for other languages in that particular language specific property file.

5. Place locale specific images in the **images** folder of the deployed **axm-jboss-agent-filterui-5.0.war** application for internationalization support for images.

   Note: For example, to create the French language specific images, you must create a folder called **fr** in the **images** folder as mentioned above and and place the french language specific images in **fr** folder and update that folder path in the locale specific property file **LogonPages_fr.properties** file.

Verifying Internationalization support

Before accessing a web page, ensure that the language you have configured in cleartrust.properties is part of accepted language list in browser , that is if you have configured for French language (fr) then browser must have **fr** in its **ACCEPT LANGUAGE** list.

If the configured language and the browser language do not match, the default language defined in the **cleartrust.agent.default_language** parameter is used.

You can also provide language precedence, if you have configured multiple languages in cleartrust.properties.

To provide language precedence, complete the following tasks:

1. Launch the web browser.
2. Go to **Tools> Options> Choose Languages**. Include the language codes that you want to localize.
   For example, if you have configured for two languages French and Spanish, then include the language code **fr** prior to **es**.

15 RSA Access Manager Agent Tools

This chapter explains the functionality of the RSA Access Manager web agent tools.

Cacheflush

Note: Before using the Cacheflush tool, set the variable **CACHEFLUSH_JAR** to <WASPROFILE>\installedApps\<node-name>\axm_cacheflush.ear\axm-was-agent-cacheflush-ejb-5.0.jar.

This Agent has seven types of caches. These caches will cache the calls made to the RSA Access Manager Authorization Server so that subsequent similar requests will be served from the cache rather than the Authorization Server. While this results in faster responses and increases the performance of the Agent, any changes made at the Authorization Server will not be reflected immediately in the Agent until the cache entries expire.

To solve this problem, the Agent includes a tool that can flush the entries in its caches. This cacheflush tool is implemented as an Enterprise Java Bean and is installed in WebSphere during the installation of the Agent. This bean can be invoked using the cacheflush shell script (UNIX machines) or batch file (Windows machines) present in the tools subdirectory of the Agent installation folder.

The cacheflush tool can be invoked with the following format:

```
cacheflush [-h WebSphere hostname] [-p WebSphere Bootstrap
```

```
port] [-t cache type] [-s]
```

-h: The Hostname or the IP of the machine where WebSphere is running. If this parameter is not specified, cacheflush defaults to localhost.

-p: The Bootstrap port number of the WebSphere server. This can be found in the following location:

Application Servers > *SERVERNAME* > Communications > Ports > *BOOTSTRAP_ADDRESS* in the administrative console.

If this parameter is not specified, cacheflush defaults to 2809.

-t: The type of cache to be flushed.

Cache type is a numeric value with the following mappings:

Value	Mapping
1	Authorization Allow Cache
2	Authorization Deny Cache
4	Groups Cache
8	User Properties Cache
16	Protected Resource Cache
32	Unprotected Resource Cache
64	Tokens Cache

Multiple cache types can be passed with the -t option by adding the cache type numeric values. For example, to flush Authorization Allow and Groups Cache, run the cacheflush tool with "-t 5", where 5 represents the sum of the Authorization Allow Cache (1), and Groups Cache (4) cache type values.

-s: Displays the current cache statistics in the Agent. When the -s option is specified, the cache is not flushed. The statistics displayed include the current size of the cache, the number of requests made, and the cache hit count.

For example:

1. To print the cache statistics of the Agent running at host1:

   ```
   Windows: cacheflush -h host1 -s
   UNIX: ./cacheflush.sh -h host1 -s
   ```

2. To flush all the caches of the Agent running at host1 and bootstrap port 2810:

   ```
   Windows: cacheflush -h host1 -p 2810
   UNIX: ./cacheflush.sh -h host1 -p 2810
   ```

3. To flush protected and unprotected resource caches of the Agent running at host1:

   ```
   Windows: cacheflush -h host1 -t 48
   UNIX: ./cacheflush.sh -h host1 -t 48
   ```

Permissions

The cacheflush application is protected by a role called "cacheadmin". Therefore, the user must be authenticated before running the cacheflush tool. By default the user name and password is prompted using a GUI logon window. This behavior can be changed by modifying the parameters in the sas.client.props file under the properties folder of WebSphere. When the synctool is run for the first time, this role is automatically created and granted to the RSA Access Manager Administrator. Only the RSA Access Manager Administrator who is configured in the synctool.properties can run the cacheflush tool.

Cacheflush in Deployment Manager

When the cacheflush tool is run in the Deployment Manager, be sure to clear the caches of all the servers managed by the Deployment Manager. You must run the cacheflush tool multiple times with -h and -p options, specifying the IP and bootstrap port numbers of all the servers that are managed by the Deployment Manager. Note that the cacheflush tool cannot be used to flush the caches of the Deployment Manager.

Note: If WebSphere is configured to use a certificate other than the one provided by WebSphere, the cacheflush tool may not be able to establish a connection to flush the cache(s). To solve this problem, update the com.ibm.ssl.* properties in the sas.client.props file in the WebSphere properties folder with the newly configured certificates. For more information about the certificate related issues, refer to the Chapter 17, “Troubleshooting.”

Synctool

This Agent includes a migration tool called synctool. This tool is capable of migrating all the security roles and their role-to-user or role-to-group mappings from the applications deployed in WebSphere to RSA Access Manager. It is a command line tool that connects to the specified WebSphere server and the RSA Access Manager Entitlements Server.

The parameters necessary for connecting the WebSphere server and the Entitlements Server can be specified in the synctool.properties file. Alternatively, the parameters can be specified in the command line. If the necessary properties are not present in the synctool.properties file, and they are not passed as command line arguments, then the user is prompted to supply the required properties. The installer fills in all the required properties during the installation of the Agent.

The synctool operates in one direction only. It queries all the existing roles and their role-to-user or role-to-group mappings from WebSphere and creates them in RSA Access Manager. It is strictly a one-way process. That is, changes made to users or groups in RSA Access Manager are not synced back to the application server. Also, the synctool only adds role-to-user and role-to-group mappings to RSA Access Manager. It never deletes role-to-user and role-to-group mappings from RSA Access Manager for an existing mapping. If the role-to-user and role-to-group mappings need to be removed, the RSA Access Manager administrator must do this manually in RSA Access Manager.

If an application has a role-to-group mapping assigned during the application assembly, and this group does not exist in RSA Access Manager, the synctool creates this group in RSA Access Manager. If an application has a role-to-user mapping assigned during the application assembly, and this user does not exist in RSA Access Manager, the synctool does not create this user in RSA Access Manager, and this mapping is omitted by the synctool.

When the synctool is run for the first time, all the applications with their roles, role-to-user, or role-to-group mappings are created in RSA Access Manager. When the synctool is run subsequently, only newly added applications with their roles and role-to-user or role-to-group mappings are created in RSA Access Manager. The synctool checks RSA Access Manager for the existence of any of the roles present in each WebSphere application. If the roles are found, the application is considered to be synced already and it is not resynced. This is done to prevent re-creating roles and role-to-user or role-to-group mappings that were deleted in RSA Access Manager when the synctool was run more than once.

You can override this default behavior and force the synctool to resync an application by passing the application name in the command line with the -apps option. Multiple applications can be resynced by passing a comma-separated list of application names.

Note: If all the roles for a previously synced application are deleted from RSA Access Manager, the application is considered a newly installed application the next time the synctool is run. Rerunning the synctool adds all of the application's role-to-user and role-to-group mappings to RSA Access Manager even if the -apps option is not specified in the command line.

If IBM WebSphere Application Server security is configured, Synctool requires administrative SOAP clients to use SSL in order to connect to WebSphere Application Server. You can set the following properties:

- **cleartrust.agent.synctool_was_truststore_file.** The truststore is used by synctool to establish the connection with Websphere. This value must be an absolute file path. It is required if cleartrust.agent.was_securityenabled is set to true.
- **cleartrust.agent.synctool_was_truststore.passphrase.** The passphrase to unlock the truststore file specified in the property cleartrust.agent.syncol_was_truststore_file. This must be encrypted using the ctencrypt tool.

- **cleartrust.agent.synctool_was_keystore_file.** The Keystore is used by synctool to establish the connection with Websphere. This value must be an absolute file path. It is required if cleartrust.agent.was_securityenabled is set to true.
- **cleartrust.agent.synctool_was_keystore.passphrase.** The passphrase to unlock the Keystore file that is specified in the property cleartrust.agent.synctool_was_keystore_file. This must be encrypted using the ctencrypt tool.

Note: Support for Special Application. WebSphere has two special applications, admin-authz and naming-authz. The user having access to these application's roles is only allowed to log on to admin console. Synctool creates role-to-user and role-to-group mapping for these applications.

Note: Installed agent with SSO Only option. If you have configured the agent with the SSO only option then there is no need to run the Synctool.

Support for Special Users

WebSphere has two special users: All Authenticated represents all the valid users present in the WebSphere user registry, and Everyone represents all the valid users of WebSphere and anonymous users. If a role is granted to Everyone, then the users do not need to authenticate to WebSphere to access the resource. If a role is granted to All Authenticated, then the users need to authenticate to WebSphere to access the resource, but the role does not need to be granted explictly to that user or to any of the users in the groups.

The Agent supports these special users. When the synctool is run for the first time, these users are created. The RSA Access Manager user corresponding to All Authenticated users is was_allauthenticated, and the RSA Access Manager user corresponding to Everyone is was_everyone.

Properties Required for Synctool

- **cleartrust.agent.server_name.** The unique WebSphere server name. The value must be the same as the cleartrust.agent.server_name value in the cleartrust.properties file. This unique server name becomes a part of all the role names created in RSA Access Manager. For the format of the role names, see "Mapping Conventions" on page 60.
- **cleartrust.agent.synctool_logdir.** The directory where the log file must be written. The value must be an absolute folder name.
- **cleartrust.agent.adapi.host.** The hostname of the RSA Access Manager Entitlements Server.
- **cleartrust.agent.adapi.port.** The port number where the Entitlements Server is listening.

- **cleartrust.agent.adapi.user_id.** The user name required to connect to the Entitlements Server. This user should have both read and write permissions to the Entitlements Server. The will be encrypted in synctool.properties.
- **cleartrust.agent.adapi.user_password.** The password used to connect to the Entitlements Server. This will be encrypted in synctool.properties.
- **cleartrust.agent.adapi.admin_role.** The Administrative Role of the user that is used to connect to the Entitlements Server.
- **cleartrust.agent.adapi.admin_group.** The Administrative Group of the user that is used to connect to the Entitlements Server.
- **cleartrust.agent.adapi.ssl.** The connection type to the Entitlements Server. Allowed values are Clear, Anon, and Auth. The value must match the api_port.use_ssl setting in eserver.conf.

Note: Configuring the Entitlements Server to use Auth mode connections causes the Administrative Console GUI application to fail. The Administrative Console application currently supports only Clear and Anon mode connections to the Entitlements Server.

- **cleartrust.agent.ssl.ca.keystore_file.** The CA Keystore used to connect to the Entitlements Server. This value must be an absolute file path. It is required if cleartrust.agent.adapi.ssl is set to Auth.
- **cleartrust.agent.ssl.ca.keystore_type.** The CA Keystore file type. This value must be either JKS for Java Key Store or PKCS12 for the PKCS #12 standard. It is required if cleartrust.agent.adapi.ssl is set to Auth.
- **cleartrust.agent.ssl.ca.keystore_passphrase.** The passphrase to unlock the CA Keystore. It is required if cleartrust.agent.adapi.ssl is set to Auth.
- **cleartrust.agent.ssl.private.keystore_file.** The Private Keystore file used to connect to the Entitlements Server. The value must be an absolute file path. It is required if AdminEnableSSL is set to Auth.
- **cleartrust.agent.ssl.private.keystore_type.** The Private Keystore file type. This value must be either JKS for Java Key Store or PKCS12 for the PKCS#12 standard. This property is required if AdminEnableSSL is set to Auth.
- **cleartrust.agent.ssl.private.keystore_passphrase.** The passphrase to unlock the Private Keystore file. It is required if AdminEnableSSL is set to Auth.
- **cleartrust.agent.ssl.private.key_alias.** The alias (Common Name/CN) of your private key in the Private Keystore. It is required if AdminEnableSSL is set to Auth.
- **cleartrust.agent.was_host.** The hostname of the WebSphere Application Server, which is the source for the sync to connect.
- **cleartrust.agent.was_port.** The SOAP connector port of the WebSphere Application Server. If the application server is a standalone server, the default port is 8880. For Deployment Manager the default port is 8879.
- **cleartrust.agent.was_securityenabled.** Specifies whether Global Security is turned on in the application server. Allowed values are True or False.

- **cleartrust.agent.was_adminname.** The user name of the WebSphere administrator. It is required if cleartrust.agent.was_securityenabled is set to True.
- **cleartrust.agent.was_adminpassword.** The password of the WebSphere administrator. It is required if cleartrust.agent.was_securityenabled is set to True.
- **cleartrust.agent.SECURITYBLOCK.** This property is read-only from the synctool.properties file.
- **cleartrust.agent.role_separator.** This separator is used as the conjunction between the actual role name in WebSphere, the application name, the servername, and the generic tag. This property is read-only from the synctool.properties file.

Command Line Options

Each of the cleartrust.agent.* properties has a corresponding command line option in short form. To get the list of command line options and their corresponding property mappings, run the synctool with the -help option or refer to synctool.properties. Note that the property name itself cannot be passed as a command line argument.

- **-help:** To get the list of command line options available.
- **-apps:** To completely resync one or more applications. Pass in a single application name or a comma-separated list of application names to be resynced.
- **-jacc:** If JACC is enabled, this command will resync all applications, including applications which are installed before installing the Agent.

Note: You only need to run the -jacc parameter once with synctool after restarting the WebSphere server. You do not need to run the synctool again to propagate a policy for a newly-installed application. A policy for a newly-installed applications is propagated to the JACC Provider when the application is deployed by the Agent.

The following examples show how to run the synctool. The first example uses all of the properties in synctool.properties. The third example overrides some of the properties in synctool.properties, and specifies new values to use. The last example shows how to use the "-apps" option to completely resync specified applications.

```
synctool.sh
(takes the properties from synctool.properties)

synctool.sh -help
(prints the Help)

synctool.sh -wash 192.168.201.69 - wasp 8887
(overrides synctool.properties)

synctool.sh –wasp 8887 –apps
“AccountManagement.ear,ct_cacheflush.ear,TravelBooking.ear”
(completely resyncs the specified applications)
```

Note: Microsoft Active Directory in Mixed Mode does not support nested groups. If you use Active Directory in Mixed Mode as the RSA Access Manager datastore, the synctool cannot create mappings for nested groups. If the synctool is run in such an environment, it displays an error message and then continues to sync the remaining applications.

Configuring Agent for Profile (CAP) Tool

The Configure Agent for Profile (CAP) tool is used to configure RSA Access Manager Agent 5.0 SP1 in the additional profiles of WebSphere Application Server. This tool is available in the Agent tools directory.

After running this tool, cleartrust.properties in the original profile (where the Agent was installed using the installer) must be copied into the new profile's properties directory. axm-was-agent-cacheflush-ear-5.0.ear and axm-was-agent-filterui-ear-5.0.ear applications in the Agent lib directory must be installed manually using the WebSphere Application Server adminconsole. Then you must restart the WebSphere Application Server.

Note: All the prompts generated by this tool have the same meaning as the prompts asked during the installation.

Pre-requisites for Running the CAP Tool:

1. Security must be disabled in the desired profile.
2. At least one profile must be configured with the RSA Access Manager agent using the installer.
3. All Profiles need to be un-configured before running the uninstaller in the main profile.

Running the CAP Tool

To configure the Agent in a new profile, the CAP Tool can be invoked as follows:

```
cap.sh -install
```

If cap.sh -install is executed with the -install option then the tool will prompt for the following:

- Profile directory location
- SOAP port number
- Admin username
- Admin password
- LTPA password

After validating the input values, the tool configures the RSA Access Manager Agent for the desired profile.

If you need to use a different server name, cleartrust.properties must be edited in the new profile before running the synctool.

The synctool needs to be executed before restarting the WebSphere Application Server for the changes to go into effect.

The CAP tool can be invoked to un-configure the Agent in the profile as follows:

```
cap.sh -uninstall
```

If cap.sh is executed with the -uninstall option, then the tool will prompt for the following:

- SOAP port number
- Security Enabled
- Admin UserName
- Admin Password
- KeyStore
- KeyStore Passphrase
- Trust Store
- Trust Store Passphrase

Ctencrypt

The Agent requires that certain confidential property values to be encrypted in the cleartrust.properties and synctool.properties file. The Agent installer automatically encrypts all the confidential property values during installation. Using the ctencrypt utility, these confidential property values can be changed and re-encrypted at a later time.

The following parameters can be encrypted:

- cleartrust.agent.adapi.user_id
- cleartrust.agent.adapi.user_password
- cleartrust.agent.adapi.user_id
- cleartrust.agent.adapi.user_password
- cleartrust.agent.ssl.ca.keystore_passphrase
- cleartrust.agent.ssl.private.keystore_passphrase
- cleartrust.agent.ssl.private.key_passphrase
- cleartrust.agent.was_adminname
- cleartrust.agent.was_adminpassword
- cleartrust.agent.synctool_was_truststore.passphrase
- cleartrust.agent.synctool_was_keystore.passphrase

For Example:

```
ctencrypt mode lockbox_file_path <item-name> param1=value1
param2=value2 param3=value3 param4=value4 param5=value5
```

where,

–***mode*** specifies whether to encrypt the value using the FIPS algorithm. Allowed values : fips or nonfips.

–***lockbox_file_path*** specifies the location where the lockbox file is stored.

–***param1*** and ***param2*** are the two parameters to encrypt.

–***value1*** and ***value2*** are the values assigned to the two parameters.

–***itemname*** the key (or item) required for all item actions. This item-name is used to associate the password in lockbox. Update the same key name in all the relevant configuration files.

Lockbox File Tool

This section describes about the steps that need to be followed to create the Lockbox file which is used by the ctencrypt tool.

The Lockbox file stores the encryption key used by ctencrypt tool. The encryption key is derived from the encryption password supplied in the arguments while creating or adding new item in the Lockbox file.

The Lockbox file securely stores secrets such as encryption keys. Each value stored in the Lockbox file is referenced by using a key item.

The Lockbox file stores data in key-value pairs. There by, one Lockbox file can be used for different components for storing their encryption keys. To Store multiple encryption keys in a Lockbox file choose different key items.

Note: If you enable the LockBox feature and run an existing Lockbox file, the utility adds the key item and encryption key to the Lockbox file.

To create and generate the Lockbox file, you must:

1. Open command prompt and navigate to **AGENTBASE/tools.**
2. Run the lockbox-tool. Type:

   ```
   lockbox-tool -passphrase <phrase> [-lockbox <filepath>]
   -[-{create|delete|exist|changepwd}] <item-name> <value>
   [-{addhost|deletehost|listhosts}] <hosts>
   ```

   The switch names can be abbreviated and are not case sensitive.

   `-passphrase <phrase>`:Lockbox Passphrase used to generate the encryption key. The passphrase must meet the following requirements.

   - 8 or more characters in length
   - Contain at least one numeric character
   - Contain at least one uppercase and lowercase character

– Contain at least one non-alphanumeric or special character such as # or !

`-lockbox <filepath>`: Path of the Lockbox file. If the file is not found, a new file will be created with the name "*lockbox.clb*". If found, the file gets updated.

`-create | -delete | -exist`: The key (or item) must be present to perform lockbox keyitem management. Create, delete or check for the existence of an item in the lockbox. It can be an arbitrary string but should not start with __.

`-changepwd`: Changes the lockbox passphrase.

`<item-name>`: The key (or item) required for all item actions. This item-name is used to associate the password in lockbox. Update the same key name in all the relevant configuration files.

`<value>`: The key (or item) required to create item action and lockbox passphrase change action. In case of create item action this value is the Password which will be used for encrypting/decrypting the configuration.

`-addhost | -deletehost | -listhosts`: The key (host actions) must be present to perform lockbox host management. Adds, deletes and prints the hostnames in the lockbox.

`<hosts>`: List of host names separated by comma in which the lockbox file is used. Required for addhost and delete host actions. Currently lockbox file is required for ctencrypt tool.

The generated Lockbox file can be further used by ctencrypt Tool, RSA Access Manager Agents or Lockbox file tool.

Install the Lockbox File Dependencies

Lockbox file relies on the applications to function properly. These applications need to be installed in those machines where Lockbox file needs to be used. The ctencrypt Tool use the Lockbox file tool.

The Lockbox file dependency has to be installed in the machines where EncryptUtil Tool is deployed or used.

To install Lockbox file on a Windows platform:

1. Navigate to the ***/cst*** directory in the installation kit.
2. Select the Windows 32 bit or Windows 64 bit CST kit.
3. Copy the CST kit to a temporary directory.
4. From the temporary location, run **vcredist.exe** to install Visual C++ 2010 redistributable package.
5. Update the PATH environment to point to the temporary directory.

To install Lockbox file on a Linux or UNIX platform:

1. Navigate to the **/cst** directory in the installation kit.
2. Copy the CST kit to a temporary directory.
3. Set **LD_LIBRARY_PATH** environment variable to point to the temporary directory.

16 Caching and Performance Tuning

This chapter explains the functionality of the RSA Access Manager web agent Caching and Performance tools.

Authorization Table Caches

The RSA authorization table uses User Groups Cache.

Trust Association Interceptor Caches

The RSA Trust Association Interceptor uses Token Cache.

Exclusion Lists

Exclusion lists improve the performance of the Agent by excluding access checks for the specified resources.

URL Exclusion List

This specifies a list of URLs to be excluded from access control checks. There is no need to specify the RSA Access Manager logon forms here, as the Agent automatically includes them in the exclusion list. The configured parameter is cleartrust.agent.uri_exclusion_list.

For example:

```
cleartrust.agent.uri_exclusion_list=/../images/
```

Extension Exclusion List

This specifies a list of file extensions. The URLs that end with these file extensions are excluded from access control checks. The configured parameter is cleartrust.agent.extension_exclusion_list.

```
For example: cleartrust.agent.extension_exclusion_list=gif,jpg
```

Caching Parameters

When a request is made to the Authorization Server, the result of the request is cached in the Agent so that the subsequent request can be served from the cache rather than the Authorization Server.

The Authorization Server and Agent caches are independent, and are not related in any way. RSA recommends enabling caching in both the Authorization Server and the Agent. There are seven different types of cache maintained in the Agent. The size and time to live property of each of these caches can be configured independently. You can disable caches independently or disable all the caches.

All the caches in the Agent can be disabled by setting the cleartrust.agent.enable_cache property to False.

There are two types of cache parameters that can be configured for each cache.

- **Cache size.** This specifies the maximum number of entries that can be cached in a single cache. When the cache table reaches the size specified by this parameter, the oldest entries are removed from the cache first. A value of zero (0) means the cache table can grow infinitely. There is no limit on the table size, but entries are deleted as they expire. If this property is not set for a cache then that particular cache is disabled.
- **Time to Live (TTL).** The value specified for this parameter is used to determine how frequently the Agent deletes the cache entries. A value of '0 Secs' means that the cache entries are never deleted based on TTL. If this property is not set for a cache then that particular cache is disabled.

The cache parameters set cache size and the amount of time that a entry is stored in the cache. Setting these parameters helps increase the performance of the Agent. The RSA Access Manager Server caches update whenever there is a change through the Administrative Console or through the Administrative API. The Agent cache is updated based on the "Time To Live" (TTL) settings.

The different types of Cache maintained in the agent are:

- Protected Resource Cache
- Unprotected Resource Cache
- Authorization Allow
- Authorization Deny
- Group
- Token
- User Properties

Protected Resource Cache

Properties:

```
cleartrust.agent.protected_resource_cache_size
cleartrust.agent.protected_resource_cache_ttl
```

Unprotected Resource Cache

Properties:

```
cleartrust.agent.unprotected_resource_cache_size
cleartrust.agent.unprotected_resource_cache_ttl
```

Authorization Allow Cache

Properties:

```
cleartrust.agent.authz_allow_cache_size
cleartrust.agent.authz_allow_cache_ttl
```

Authorization Deny Cache

Properties:

```
cleartrust.agent.authz_deny_cache_ttl
cleartrust.agent.authz_deny_cache_size
```

Group Cache

Properties:

```
cleartrust.agent.groups_cache_ttl
cleartrust.agent.groups_cache_size
```

Token Cache

Properties:

```
cleartrust.agent.token_cache_ttl
cleartrust.agent.token_cache_size
```

User Properties Cache

Properties:

```
cleartrust.agent.userprops_cache_ttl
cleartrust.agent.userprops_cache_size
```

17 Troubleshooting

This section describes possible solutions to some of the problems that you may encounter during or after the configuration and deployment of the Agent.

Incorrect Agent behavior when the agent is configured with Servers in Passive mode

To protect an application in the passive mode, do the following:

1. Provide the entitlements for all the CT logon pages to the administrator user, the was_everyone user, and the was_allauthenticated user.
2. Protect </application_name/*> and </application_name.war/*> with Access Manager.

Note: Ensure that the entitlements for all the CT logon pages are given to the users mentioned in step 1.

When you access the resource, the Websphere Application Server (WAS) security alert appears. After you authenticate to the WAS default realm, the CT challenge page appears. Login with your RSA Access Manager Agent 5.0 SP1 credentials to access the resource.

Incorrect User name or Password Causes Agent or Synctool to Fail

To remove errors caused by incorrect User name or password, do the following:

- WebSphere fails to start. SystemOut.log file shows:

  ```
  checkPassword(): Password Check Failed Reason
  :INVALID_PASSWORD
  checkPassword(): Password Check Failed Reason :UNKNOWN_USER
  ```

- Synctool fails to communicate to Entitlements Server due to Unknown User or Wrong Password Error.
- Synctool fails to communicate to WebSphere Admin Port due to Unknown User or Wrong Password Error.

These errors occur if the user name or password given during the installation is invalid, or if the password of the WebSphere administrative user has expired.

During the Agent's installation, the installer does not verify the user names and their passwords. Therefore it is possible to install the Agent with a incorrect user name or password.

Three sets of user names and passwords are configured during Agent installation.

- WebSphere Administrative User

  To change this user or the user's password, follow these steps:

a. Open the security.xml file under the WASPROFILE/config/cells/CELL_NAME folder.

b. Search for the string userRegistries xmi:type=security:CustomUserRegistry. Within this tag edit the "serverId" and "serverPassword" values. These are the user name and password that are configured as WebSphere Administrator. Replace these values with the correct ones. The server password is usually encoded using XOR. You can simply replace the server password value with the plain text value.

c. To encode the password again, start WebSphere and log on to the administrative console.

- In WebSphere 6.0.2, Click Security > User Registries, then select Custom.
- In WebSphere 6.1.0.3, Click Security > Secure administration, applications, and infrastructure, then select Standalone custom registry from Available realm definition and click Configure, then click OK.
- In WebSphere 7.0, Click Security > Global Security, then select Standalone custom registry from Available realm definition and click Configure, then click OK.

d. WebSphere prompts you to save the configuration. When you save the configuration, the password is XOR encoded again.

- RSA Access Manager Administrative User in cleartrust.properties File

  To change this user or the user's password, run the ctencrypt utility located in the /tools folder:

  – Windows: ctencrypt securityblock=cleartrust.agent.SECURITYBLOCK value from cleartrust.properties user= newuser password= newpassword

  – UNIX: ctencrypt.sh securityblock=cleartrust.agent.SECURITYBLOCK value from cleartrust.properties user=newuser password=newpassword

  Copy the encrypted values displayed on the screen to the cleartrust.properties file.

- RSA Access Manager Administrative User in synctool.properties File

  To change this user or the user's password, run the ctencrypt utility present under the AGENT-INSTALLATION-DIR/tools folder:

```
Windows: ctencrypt
securityblock=cleartrust.agent.SECURITYBLOCK value from
synctool.properties user=newuser password=newpassword
UNIX: ctencrypt.sh
securityblock=cleartrust.agent.SECURITYBLOCK value from
synctool.properties user=newuser password=newpassword
```

  Copy the encrypted value displayed on the screen to the synctool.properties file.

Unable to Stop WebSphere when RSA Access Manager Servers are not running

When the "stopserver" script is run to stop the WebSphere server, and the RSA Access Manager Servers are not running, the script exits with an error saying that authentication or authorization failed for the user.

This is the intended behavior. The RSA Access Manager Servers must be running when you attempt to stop the WebSphere server.

Only WebSphere administrators are allowed to stop the server. The "stopserver" script tries to authenticate first with the WebSphere server. If the RSA Access Manager Servers are not running, WebSphere cannot authenticate the user and fails the request to stop the server.

javax.net.ssl.* Errors when Using Certificate Generated by RSA Keon Certificate Authority

IBM WebSphere has problems handling a certificate generated by RSA Keon Certificate Authority when the certificate has the "key usage" field marked as "critical". This is a known limitation. To solve this problem, do not set the "key usage" field as "critical" in the certificate.

You encounter this problem

- While installing or reinstalling the Agent with Global Security enabled.
- When running the cacheflush tool, if WebSphere is configured to use a certificate generated by RSA Keon Certificate Authority.

Protecting Web Resources at Both Front-End Web Server and WebSphere Using Web Filter Can Cause Authentication Failure

When a resource protected with web filter is accessed through a web server, and if the resource is also protected in the front-end web server, web filter fails the request. This happens when the front-end web server authenticates the user, the IBM WebSphere plug-in inserts a BASIC authentication header with user name and empty password into the request and forwards it to WebSphere. Web filter tries to authenticate the user using the BASIC authentication header in the request, and fails due to the wrong password. This happens when form-based authentication is disabled in cleartrust.properties.

To solve this problem, do not protect the resource at both ends, or use form-based authentication at both ends.

Problem Using the Agent with RSA ClearTrust Server 5.0.1 Using Mutually Authenticated SSL Between the Agent and the RSA ClearTrust Server

When the Agent is configured in authenticated SSL mode with RSA ClearTrust Server 5.0.1, the Agent fails to communicate with the RSA ClearTrust Servers, giving certificate-related errors. Using mutually authenticated SSL connection mode is not supported by this version of the Agent.

Login Unsuccessful when RSA Web Filter-Protected Resource Is Accessed Through SSL (HTTPS)

When an RSA web filter-protected resource is accessed through HTTPS using Microsoft Internet Explorer, the user is prompted repeatedly for authentication. This happens if the client machine is updated with Internet Explorer security update Q832894 or hotfix 821814. These Microsoft updates have a bug which causes the browser to send a POST request with empty POST data to the RSA Access Manager Agent.

Microsoft has issued a fix for this bug. The client can download the appropriate cumulative security update for Internet Explorer from http://www.microsoft.com/technet/security/Bulletin/MS04-004.mspx.

Unable to Protect Only the POST Method of a Web Resource

WebSphere does not protect role-based web applications that have method-level protection of only the POST method. To protect individual methods on HTTP requests such as GET and POST, and provide authorization to resources at this level of granularity, both the GET and POST methods must be protected. Otherwise, the authentication and authorization process does not work correctly. This is a WebSphere limitation.

Authentication Changes Require Restarting JMS Server when Protecting Message Driven Beans

The Agent provides container-managed authentication support for Message Driven Beans. For container-managed authentication, WebSphere authenticates the user ID with the Agent during WebSphere startup, and stores the credential internally. This stored credential is used by WebSphere whenever Message Driven Beans are accessed. Any change in the authentication data, for example, a password change, expired password, admin lockout, and so on, is not reflected until the JMS server is restarted. This behavior is by design. You must restart the JMS server for the changes to take effect.

Cacheflush Error when Running WebSphere in SSL Mode

When WebSphere is running in SSL mode, cacheflush.sh might throw the following error:

```
JSAS1477W: SECURITY CLIENT/SERVER CONFIG MISMATCH: The client
security configuration (sas.client.props or outbound settings
in GUI) does not support the server security configuration for
the following reasons:

ERROR 1: JSAS0606E: The server requires SSL client certificate
authentication but the client does not support it.
```

The most likely reason for this error is that the client security settings do not match the server security settings. Check the sas.client.props file in WAS-HOME/DeploymentManager/properties or WAS-HOME/AppServer/properties to ensure that the following parameter settings match the server settings:

- -com.ibm.CSI.performTLClientAuthenticationRequired
- -com.ibm.CSI.performTLClientAuthenticationSupported

For example, if the server is set to "support client certificate" and not "require client certificate", set the parameters as follows:

```
com.ibm.CSI.performTLClientAuthenticationRequired=false
com.ibm.CSI.performTLClientAuthenticationSupported=true
```

Using Multiple Virtual Hosts with Web Filter

WebSphere supports the concept of virtual hosts. You can deploy applications to different virtual hosts, but these applications are served out of the same physical server. For more information on virtual hosts, see the WebSphere documentation.

To protect applications on different virtual hosts using the Web Filter, the RSA Access Manager application, and cleartrust.ear must be mapped to each virtual host. To do this, deploy the cleartrust.ear application to each virtual host defined in your environment, and change the application name for cleartrust.ear on each virtual host as you deploy. You must change the application name for each virtual host, because WebSphere does not allow duplicate applications to be deployed on the same server instance.

The preceding configuration step is mandatory. Otherwise, when a user tries to access the protected application, the RSA Access Manager Web Filter tries to redirect the user to the login page, but since the cleartrust.ear file is not deployed on that virtual host, it results in "404: Page Not found" errors.