

Policy Agent Guide

Sun[™] ONE Identity Server

Version 6.0

816-6688-10
March 2003

Copyright © 2003 Sun Microsystems, Inc. All rights reserved.

Sun, Sun Microsystems, and the Sun logo are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Federal Acquisitions: Commercial Software -- Government Users Subject to Standard License Terms and Conditions. The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation. No part of the product or this document may be reproduced in any form by any means without prior written authorization of the Sun Microsystems, Inc. and its licensors, if any.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Some preexisting portions Copyright (c) 1999 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

"This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)."

Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.
5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2003 Sun Microsystems, Inc. Tous droits réservés.

Sun, Sun Microsystems, le Sun logo, et iPlanet sont des marques dposes ou des marques dposes registre de Sun Microsystems, Inc. aux Etats-Unis et d'autres pays. Le produit dé crit dans ce document est distribué selon des conditions de licence qui en restreignent l'utilisation, la copie, la distribution et la décompilation. Aucune partie de ce produit ni de ce document ne peut être reproduite sous quelque forme ou par quelque moyen que ce soit sans l'autorisation écrite préalable de Sun Microsystems, Inc., le cas échéant, de ses bailleurs de licence.

CETTE DOCUMENTATION EST FOURNIE "EN L'ÉTAT", ET TOUTES CONDITIONS EXPRESSES OU IMPLICITES, TOUTES REPRÉSENTATIONS ET TOUTES GARANTIES, Y COMPRIS TOUTE GARANTIE IMPLICITE D'APTITUDE À LA VENTE, OU À UN BUT PARTICULIER OU DE NON CONTREFAÇON SONT EXCLUES, EXCEPTÉ DANS LA MESURE OÙ DE TELLES EXCLUSIONS SERAIENT CONTRAIRES À LA LOI.

Contents

About This Guide	11
What You Are Expected to Know	11
Identity Server Documentation Set	12
What's in This Guide	12
Documentation Conventions Used in This Manual	13
Typographic Conventions	13
Terminology	14
Related Information	14
Part 1 Web and Proxy Agents	17
Chapter 1 Read This First	19
How Policy Agents Work	19
Uses for Policy Agents	19
How an Agent Interacts with Sun ONE Identity Server 6.0	20
Supported Servers	21
Before You Begin Installation	22
Java Runtime Environment 1.3.1	23
The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers	23
Configuring Agent for Multiple Web Server Instances on the Same Computer System	23
Providing Failover Protection for Sun ONE Identity Server Agents	24
Updating the Agent Cache	24
Global Not-Enforced URL List	25
Global Not-Enforced IP Address List	26
Enforcing Authentication Only Without Enforcing Policies	27
Forwarding LDAP User Attributes via HTTP Headers	27

The AMAgent.properties File	29
Setting the Fully Qualified Domain Name	30
Cookie Reset Feature	32
Configuring CDSSO	32
Verifying a Successful Installation	33
 Chapter 2 Policy Agents for Solaris 8 and 9	35
Before You Begin	36
Supported Solaris Web Servers	36
Patch Cluster for Solaris	37
Installation Using the GUI	37
Installing a Proxy Server Policy Agent	41
Installation Using the Command-Line	43
To Install a Web Server Agent Using the Command-Line	43
To Install a Web Proxy Server Agent Using the Command-Line	45
Configuring Agent for Multiple Web Server Instances	47
To Configure Agent for Multiple Web Server Instances on the Same Computer System	48
Using the config Script for Silent Installations	49
Removing an Agent Using the unconfig Script	51
Using Secure Sockets Layer (SSL) with an Agent	51
Configuring the IBM HTTP Server	52
Configuring the Domino DSAPI Filter	52
Web or Web Proxy Server Running in SSL Mode	53
The Agent's Default Trust Behavior	53
Disabling the Agent's Default Trust Behavior	53
Installing the Root CA Certificate on the Remote Web Server	54
Setting the REMOTE_USER Server Variable	59
Validating Client IP Addresses	59
POST Data Preservation	60
Shared Secret Encryption Utility	60
Uninstalling a Policy Agent	61
Before Uninstalling the Policy Agent for Lotus Domino 5.0.10	61
Uninstalling Using the GUI	62
Uninstalling Using the Command-Line Interface	62
Troubleshooting Solaris Agents	63
Known Problems	66
 Chapter 3 Policy Agents for Windows 2000	69
Before You Begin	69
Supported Windows Web Servers	70
Installing the Policy Agent for Microsoft IIS	70
Installation Using the GUI	71

Installation Using the Command-Line	74
Installing an Agent Using the Command-Line	74
Configuring the Domino DSAPI Filter	76
Configuring Domino DSAPI Filter for Multiple Server Partitions	77
Using Secure Sockets Layer (SSL) with an Agent	77
The Agent's Default Trust Behavior	78
Disabling the Agent's Default Trust Behavior	78
Installing the Root CA Certificate on the Remote Web Server	79
Setting the REMOTE_USER Server Variable	81
Validating Client IP Addresses	81
POST Data Preservation	82
Shared Secret Encryption Utility	83
Uninstalling and Disabling Policy Agent	83
Uninstalling a Policy Agent	83
Disabling a Policy Agent Installed on Microsoft IIS	84
Uninstalling the Policy Agent for Lotus Domino 5.0.10	84
Uninstalling an Agent Using the Command-Line	85
Troubleshooting the Installation	86
IIS Policy Agent	86
Known Problems	91
 Chapter 4 Policy Agents for Windows NT	93
Before You Begin	93
Supported Windows NT Web Server	94
Installation Using the GUI	94
Installing the Policy Agent for Microsoft IIS 4.0	94
Uninstalling and Disabling Policy Agents	97
Installation Using the Command-Line	98
To Install an Agent Using the Command Line	98
To Uninstall an Agent Using the Command Line	100
Using Secure Sockets Layer (SSL) with an Agent	101
The Agent's Default Trust Behavior	102
Disabling the Agent's Default Trust Behavior	102
Installing the Root CA Certificate on the Remote Web Server	102
Setting the REMOTE_USER Server Variable	103
Validating Client IP Addresses	104
Shared Secret Encryption Utility	105
Troubleshooting the IIS 4.0 Policy Agent	105
Known Problems	110
 Chapter 5 Policy Agent for Red Hat Linux 7.2	113
Before You Begin	113

Configuring Apache Web Server with Posix Threads	114
Installation Using the GUI	115
Installing the Policy Agent	115
Uninstalling the Policy Agent	118
Installation Using the Command-Line	118
Installing the Policy Agent	118
Uninstalling the Policy Agent	120
Configuring Agent for Multiple Web Server Instances	122
To Configure Agent for Multiple Web Server Instances on the Same Computer System	122
Using the config Script for Silent Installations	123
Removing an Agent Using the unconfig Script	125
Using Secure Sockets Layer (SSL) with an Agent	125
The Agent's Default Trust Behavior	125
Disabling the Agent's Default Trust Behavior	126
Installing the Root CA Certificate on the Remote Web Server	126
Setting the REMOTE_USER Server Variable	127
Validating Client IP Addresses	128
Shared Secret Encryption Utility	128
Troubleshooting Information	129

Part 2 J2EE Agents 131

Chapter 6 Read This First	133
Uses of Policy Agent for Application Server	133
Examples	134
Supported Servers	135
 Chapter 7 Policy Agent for WebLogic 6.1 SP2	 137
Supported Platforms	137
How the Policy Agent for WebLogic 6.1 SP2 Works	138
Guidelines	138
Installing the Agent	139
Pre-installation Tasks	139
Launching the Installation Program on Solaris 8	140
Launching the Installation Program on Windows 2000 Server	142
Launching the Installation Program on HP-UX 11	143
Installing the Agent Using GUI	145
WebLogic Server Configuration	155
Installing the Agent Realm	155
Troubleshooting the Installation	158
Application Configuration	159

Installing the Agent Filter Component in an Application	159
Creating Role-to-Principal Mappings	161
Application Specific Agent Configuration	161
Special Case: Default Web Application	164
Global Agent Configuration	165
Not-Enforced List Usage Considerations	165
Agent Configuration	166
Common Configuration	167
Audit Configuration	168
Realm Configuration	169
Global Filter Configuration	172
Application Filter Configuration	180
Debug Engine Configuration	183
Using Agent and Sun ONE Identity Server SDK APIs	186
Uninstalling the Agent	187
Uninstalling the Agent Using GUI	195
Troubleshooting Uninstallation Problems	195
 Chapter 8 Policy Agent 2.0 for IBM WebSphere 4.0.4 AE	197
Overview	197
Guidelines	199
Agent-based Authentication and Authorization	200
Coarse Grained and Fine Grained Access-control	200
Create Enhanced Security Aware Applications	201
Limitations	201
Supported Platform	202
Software Requirements	202
Installing the Agent	203
Pre-Installation Tasks	203
Launching the Installation Program	206
Installing the Agent Using GUI	207
Agent Configuration Details	214
Installing the Agent Using Command-Line	216
Configuring WebSphere Application Server	222
Application Configuration	224
Creating Role-to-Principal Mappings	224
Application-Specific Agent Configuration	225
Providing Application-Specific Not-Enforced List	225
Special Case: Default Web Application	226
Global Agent Configuration	227
Not-Enforced List Usage Considerations	227
Agent Configuration	228
Common Configuration	229

Realm Configuration	231
Global Interceptor Configuration	234
Application Interceptor Configuration	238
Debug Engine Configuration	239
Using Agent and Sun ONE Identity Server SDK APIs	242
SSL Configuration	244
Configuration Changes in AmConfig.properties	244
Configuration Changes in serverconfig.xml	245
Importing the root CA cert into Application Server JVM Keystore	245
Uninstalling the Agent	246
Launching the Uninstallation Program	246
Uninstalling the Agent Using GUI	247
Uninstalling the Agent Using Command Line	248
Troubleshooting Information	249
Installation/Uninstallation Problems	249
Fixing the Problems Manually	250
 Chapter 9 Policy Agent for Sun ONE Application Server 7.0	251
Supported Platforms	251
Guidelines	252
Installing the Agent	252
Pre-installation Tasks	252
Installing the Agent Using GUI	256
Installing the Agent Using Command-Line	266
Silent Installation	270
Application Configuration	272
Installing the Agent Filter Component in an Application	272
Creating Role-to-Principal Mapping	274
Application-Specific Agent Configuration	274
Special Case: Default Web Application	276
Agent Configuration	279
Common Configuration	279
Audit Configuration	281
Realm Configuration	283
Global Filter Configuration	285
Application Filter Configuration	293
Debug Engine Configuration	296
Using Agent and Sun ONE Identity Server SDK APIs	299
Uninstalling the Agent	300
Uninstalling the Agent Using GUI	303
Uninstalling the Agent Using Command-Line	304

Appendix A Configuration Tasks Performed by Installer	307
WebLogic 6.1 SP2	307
WebLogic Server Startup Script Modifications	307
Adding Parameters to Java Virtual Machine	310
Installation of JCE 1.2.1 and JSSE 1.0.2 Extensions	311
WebSphere 4.0.4 AE	311
Modifications to Admin Server Configuration File	311
Modifications to trustedserver.properties	312
Modifications to sas.client.props	312
Configurations Through Administrative Console	312
Agent Realm Configuration	313
Sun ONE Application Server 7.0	315
Application Server Config Files	315
 Appendix B Sample Scenarios for Role-to-Principal Mapping	 319
WebLogic 6.1 SP2	319
Declarative Security	319
Programmatic Security	320
WebSphere 4.0.4 AE	322
Web Authorization	322
EJB Authorization	323
Sun ONE Application Server 7.0	325
Declarative Security	325
Programmatic Security	326
 Appendix C Using the Policy Agent Debug Engine	 329
 Index	 331

About This Guide

This Policy Agent Guide offers an introduction to Sun™ ONE Identity Server Policy Agent and describes how to install and configure Sun ONE Identity Server Policy Agent on Web Servers, Proxy Servers, and Application Servers.

This preface contains the following sections:

- What You Are Expected to Know
- Identity Server Documentation Set
- What's in This Guide
- Documentation Conventions Used in This Manual
- Related Information

What You Are Expected to Know

This book is considered to be an auxiliary manual in the documentation series provided with Sun ONE Identity Server 6.0. It's essential that you understand directory technologies and have some experience with Java and XML programming languages. You will get the most out of this guide if you are familiar with directory servers and Lightweight Directory Access Protocol (LDAP). Particularly, you should be familiar with Sun ONE Directory Server and the documentation provided with that product.

This guide is intended for use by IT professionals who manage access to their network through Sun ONE servers and services. The functionality contained in Sun ONE Identity Server allows you to manage user data and enforce access policies throughout your enterprise.

As you try to understand the concepts described in this guide, you should reference the *Sun ONE Identity Server Installation and Configuration Guide* and the *Sun ONE Identity Server Programmer's Guide*.

Identity Server Documentation Set

The Sun ONE Identity Server documentation set contains the following titles:

- *Product Brief* provides an overview of the Sun ONE Identity Server application and its features and functions.
- *Installation Guide* provides details on how to install and deploy the Identity Server on Solaris™, Linux and Windows® 2000 systems.
- *Administration Guide* describes how to use the Identity Server console as well as manage user and service data via the command line.
- *Programmer's Guide* documents how to customize an Identity Server system specific to your organization. It also includes instructions on how to augment the application with new services using the public APIs.
- *Policy Agent Guide* (this guide) documents how to install and configure an Identity Server policy agent on a remote server. It also includes troubleshooting and information specific to each agent.
- *Getting Started Guide* documents how to use various features of the Identity Server product to set up a simple organization with identities, policies and roles.
- The *Release Notes* file gathers an assortment of last-minute information, including a description of what is new in this release, known problems and limitations, installation notes, and how to report problems.

NOTE Be sure to check the Identity Server documentation web site for updates to the release notes and for revisions to the guides. They are available at <http://docs.sun.com/db/prod/slidsrv#hic>. Updated documents will be marked with a revision date.

What's in This Guide

The following table lists all the Agents documented in this Guide.

Table 1 Agents Documented in This Guide

Agents	Platforms
Sun ONE Web Server 6.0 SPx	Solaris 8
Sun ONE Web Server 4.1 SP8	Solaris 8

Table 1 Agents Documented in This Guide

Agents	Platforms
Sun ONE Web Proxy Server 3.6 (in reverse proxy mode)	Solaris 8
Microsoft IIS 5.0	Windows 2000
Sun ONE Web Server 6.0 SPx	Windows 2000
Microsoft IIS 4.0	Windows NT 4.0
Sun ONE Web Server 6.0 SPx	Solaris 9
Apache 1.3.26	Solaris 8
Apache 1.3.26	Solaris 9
Apache 1.3.26	Red Hat Linux 7.2
IBM HTTP Server 1.3.19	Solaris 8
IBM WebSphere 4.0.4 AE	Solaris 8
Lotus Domino 5.0.10	Solaris 8
Lotus Domino 5.0.10	Windows 2000
WebLogic 6.1 SP2	Solaris 8
WebLogic 6.1 SP2	Windows 2000
WebLogic 6.1 SP2	HP-UX 11
Sun ONE Application Server 7.0	Solaris 8
Sun ONE Application Server 7.0	Solaris 9
Sun ONE Application Server 7.0	Windows 2000

Documentation Conventions Used in This Manual

In the Sun ONE Identity Server 6.0 documentation set (such as this guide) there are certain typographic and terminology conventions used to simplify discussion and to help you better understand the material. These conventions are described below.

Typographic Conventions

This book uses the following typographic conventions:

- *Italic type* is used within text for book titles, new terminology, emphasis, and words used in the literal sense.

- *Monospace font* is used for sample code and code listings, API and language elements (such as function names and class names), filenames, pathnames, directory names, HTML tags, and any text that must be typed on the screen.
- *Italic serif font* is used within code and code fragments to indicate variable placeholders. For example, the following command uses *filename* as a variable placeholder for an argument to the gunzip command:

```
gunzip -d filename.tar.gz
```

Terminology

Below is a list of the general terms that are used in the Sun ONE Identity Server Policy Agent documentation:

- *Agent_Install_Dir* is a variable placeholder for the directory where you have installed the Sun ONE Identity Server Policy Agent.
- *S1IS_Install_Dir* is a variable placeholder for the home directory where you have installed Sun ONE Identity Server 6.0.
- *WebLogic_Install_Dir* is a variable placeholder for the home directory where you have installed WebLogic Server.

Related Information

In addition to the documentation provided with Sun ONE Identity Server, there are several other sets of documentation that might be helpful. This section lists these and additional sources of information.

iPlanet Directory Server Documentation

iPlanet Directory Server 5.1 documentation can be found at

http://docs.sun.com/db/coll/S1_ipDirectoryServer_51

iPlanet/Sun ONE Web Server Documentation

iPlanet/Sun ONE Web Server documentation can be found at

http://docs.sun.com/db/coll/S1_ipwebsrvree60_en

Sun ONE Certificate Server Documentation

Sun ONE Certificate Server documentation can be found at

http://docs.sun.com/db/coll/S1_s1CertificateServer_47.

iPlanet Proxy Server Documentation

iPlanet Proxy Server documentation can be found at

http://docs.sun.com/db/coll/S1_ipwebproxysrvr36.

Other iPlanet Product Documentation

Documentation for all other Sun ONE servers and technologies can be found at

<http://docs.sun.com/prod/ds/sunone>

Download Center

Links to download any of Sun's Sun ONE/iPlanet software are at

<http://www.sun.com/software/download/>.

Sun ONE Technical Support

Technical Support can be contacted through

<http://www.sun.com/service/support/software/ipplanet/index.html>

Professional Services Information

Professional Service can be contacted through

<http://www.sun.com/service/sunps/ipplanet/>

Sun Enterprise Services for Solaris Patches And Support

Solaris patches and support can be obtained through

<http://www.sun.com/service/>

Developer Information

Information on Sun[™] ONE Identity Server, LDAP, the Sun ONE Directory Server, and associated technologies can also be found at

<http://developer.ipplanet.com/tech/directory/>

Related Information

Web and Proxy Agents

Chapter 1, “Read This First”

Chapter 2, “Policy Agents for Solaris 8 and 9”

Chapter 3, “Policy Agents for Windows 2000”

Chapter 4, “Policy Agents for Windows NT”

Chapter 5, “Policy Agent for Red Hat Linux 7.2”

Read This First

This chapter provides a brief overview of Sun ONE Identity Server Policy Agents for web and web proxy servers, as well as some concepts you will need to understand before proceeding with the Installation program. The information in this chapter is common to Solaris, Windows and Linux operating systems.

Topics include:

- How Policy Agents Work
- Supported Servers
- Before You Begin Installation

How Policy Agents Work

Sun ONE Identity Server Policy Agents protect content on Web Servers and Web Proxy Servers from unauthorized intrusions. They control access to services and web resources based on the policies configured by an administrator.

Uses for Policy Agents

Policy agents are installed on web servers for a variety of reasons. Here are three examples:

- An agent on a Human Resources server prevents non-Human Resources personnel from viewing confidential salary information and other sensitive data.
- An agent on an Operations web server allows only network administrators to view network status reports or to modify network administration records.

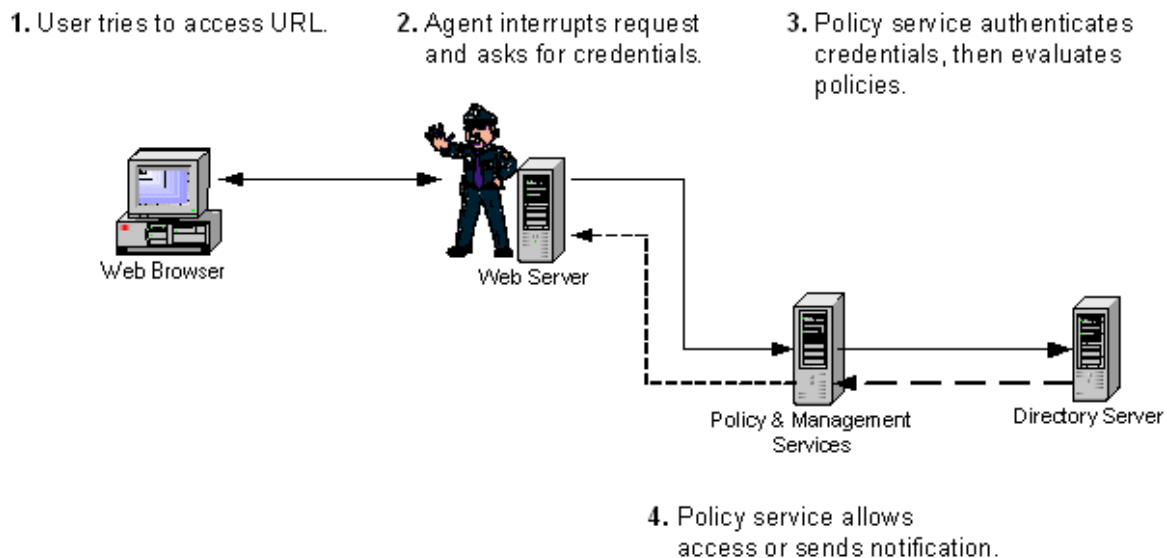
- An agent on an Engineering web server allows authorized personnel from many internal segments of a company to publish and share research and development information. At the same time, the agent restricts external partners from gaining access to the proprietary information.

In each of these situations, a system administrator must set up policies that allow or deny users access to content on a web server. For information on setting policies and for assigning roles and policies to users, see the *Sun ONE Identity Server Administration Guide*.

How an Agent Interacts with Sun ONE Identity Server 6.0

Figure 1-1 illustrates how a Policy Agent installed on a remote web server interacts with Sun ONE Identity Server. When a user points a browser to a particular URL on a protected web server, the following interactions take place:

1. The agent intercepts the request and validates the existing authentication credentials. If the existing authentication level is insufficient, the appropriate Sun ONE Identity Server authentication service will present a login page. The login page prompts the user for credentials such as username and password.
2. The authentication service verifies that the user credentials are valid. For example, the default LDAP authentication service verifies that the username and password are stored in Sun ONE Directory Server. You might use other authentication modules such as RADIUS and Certificate modules. In such cases, credentials are not verified by Directory Server but are verified by the appropriate authentication module.
3. If the user's credentials are properly authenticated, the policy agent examines all the roles assigned to the user.
4. Based on the aggregate of all policies assigned to the user, the individual is either allowed or denied access to the URL.

Figure 1-1 An Agent's Interaction With Sun ONE Identity Server

Supported Servers

Sun ONE Identity Server Policy Agents support the following servers running on the Solaris 8 operating system:

- Sun ONE Web Server 6.0 SPx
- Sun ONE Web Server 4.1 SP8
- Sun ONE Web Proxy Server 3.6 (in reverse proxy mode)
- Apache 1.3.26
- IBM HTTP Server 1.3.19
- Lotus Domino 5.0.10

Sun ONE Identity Server Policy Agents support the following servers running on the Solaris 9 operating system:

- Sun ONE Web Server 6.0 SPx
- Apache 1.3.26

Sun ONE Identity Server Policy Agents support the following server running on Red Hat Linux 7.2 operating system:

- Apache 1.3.26

Sun ONE Identity Server Policy Agents support the following servers running on the Windows 2000 operating system:

- Microsoft IIS 5.0
- Sun ONE Web Server 6.0 SPx
- Lotus Domino 5.0.10

Sun ONE Identity Server Policy Agents support the following server running on Windows NT 4.0 Service Pack 6:

- Microsoft IIS 4.0

Before You Begin Installation

You should be familiar with the following concepts before you start the Installation program:

- Java Runtime Environment 1.3.1
- The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers
- Configuring Agent for Multiple Web Server Instances on the Same Computer System
- Providing Failover Protection for Sun ONE Identity Server Agents
- Updating the Agent Cache
- Global Not-Enforced URL List
- Global Not-Enforced IP Address List
- Enforcing Authentication Only Without Enforcing Policies
- Forwarding LDAP User Attributes via HTTP Headers
- The AMAgent.properties File
- Setting the Fully Qualified Domain Name
- Cookie Reset Feature

- Configuring CDSSO
- Verifying a Successful Installation

Java Runtime Environment 1.3.1

You must have the Java Runtime Environment (JRE) 1.3.1 installed or available on a shared file system in order to run the graphical user interface (GUI) version of the Agent Installation program. Currently, JRE 1.3.1 or any version higher is certified for use with the Agent Installation program. See “Installation Using the Command-Line,” on page 43 for more information.

If you are running the Windows operating system, the Installation program will install JRE 1.3.1 if it is not detected.

The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers

You can use the Installation program to install a policy agent on the web server where Sun ONE Identity Server is installed. In Sun ONE documentation, this server is referred to as the *web server that runs the Sun ONE Identity Server 6.0*. You can also use the Installation program to install additional policy agents on remote web servers in your enterprise. A *remote* web server in a Sun ONE Identity Server deployment is any web server other than the one that runs Sun ONE Identity Server. It is “remote” relative to the Sun ONE Identity Server’s dedicated web server.

Configuring Agent for Multiple Web Server Instances on the Same Computer System

If you have multiple Web servers or Proxy servers installed on one computer system, you can install a different agent for each server or server instance.

For more information, see “Configuring Agent for Multiple Web Server Instances.”

NOTE	Only one instance of Microsoft IIS server can be installed per computer system, you cannot install multiple Microsoft IIS agents on the same computer system.
-------------	---

Providing Failover Protection for Sun ONE Identity Server Agents

When you install a Policy Agent, you can specify a *failover* or backup web server for running the Sun ONE Identity Server. This is essentially a high availability option. It ensures that if the web server that runs Sun ONE Identity Server service becomes unavailable, the agent can still process access requests through the secondary or failover web server running Sun ONE Identity Server service.

To set up failover protection for the policy agent, you must first install two different instances of Sun ONE Identity Server on two separate web servers. See the detailed instructions in the *Sun ONE Identity Server Installation and Configuration Guide*, then follow the instructions in the subsequent sections of this guide to install the appropriate Agent. The agent Installation program will prompt you for the host name and port number of the failover web server you configured to work with Sun ONE Identity Server. The following property in the `AMAgent.properties` file stores the failover server name and port:

```
com.sun.am.policy.am.loginURL= http://primary_Identity
_Server.siroe.com:58080/amserver/UI/Login http://failover_Identity
_Server.siroe.com:58080/amserver/UI/Login
```

The failover server name is configurable after it has been set during the installation. It is the second entry in this property following the primary Sun ONE Identity Server login URL separated by a space.

NOTE The protocol (for example, `http` or `https`) must be the same for both primary web server and the failover web server.

Updating the Agent Cache

Each agent maintains a cache that stores the policies for every user's session. The cache can be updated by either a cache expiration mechanism or notification mechanism.

Cache Updates

Agent maintains a cache of all active sessions. Once an entry is added to the cache, it remains valid for a period of time after which the entry is considered expired and later purged.

The property `com.sun.am.policy.am.cacheEntryLifeTime` in `AMAgent.properties` determines the number of minutes an entry will remain alive in the agent cache. Once the interval specified by this property has elapsed, the entry is dropped from the cache. By default, the expiration time is set to three minutes.

Hybrid Cache Updates

In this mode, cache entry expiration still applies. In addition, the agent gets notified by the Sun ONE Identity Server service about session changes. Session changes include events such as session log-out or a session time-out. When notified of a session or policy change, the agent updates the corresponding entry in the cache. Apart from session updates, agents may also receive policy change updates. Policy changes include events such as a updating policy, deletion, or creation.

Sun ONE Identity Server Policy Agents will have the hybrid cache update mode switched on by default. This is triggered by the property `com.sun.am.policy.am.notificationEnabled` and is set to `true` in the `AMAgent.properties` file. When the property is set to `false`, the agent updates its cache through cache entry expiration mechanism only.

Restrictions due to firewalls, as well as the type of web server in use, may not allow notifications to work in certain situations. In such cases, cache entry expiration is the only way an agent can update its cache.

NOTE The notification support is not available in the following instances:

- If IIS 4.0 or IIS 5.0 is using HTTPS
 - For Apache 1.3.26 Agents on all platforms
-

Global Not-Enforced URL List

The *global not-enforced URL list* defines the resources that should not have any policies (either allow or deny) associated with them.

By default, policy agent denies access to all resources for the web server that it protects. However, various resources available through a web server (such as a web site or application) might not need to have any policy enforced. Common examples of such resources include the HTML pages and `.gif` images found in the home pages for a web site. The user should be able to browse such pages without

authenticating. These resources need to be on the global not-enforced URL list. The property `com.sun.am.policy.agents.notenforcedList` will be used for this purpose. Wild cards can be used to define a pattern of URLs. Space is the separator between the URLs mentioned in the list.

The reverse scenario would be when all resources for the web server are open to any users except a list of URLs. In that case, the property `com.sun.am.policy.agents.reverse_the_meaning_of_notenforcedList` would be used to reverse the meaning of `com.sun.am.policy.agents.notenforcedList`. If it is set to `true` (by default it is set to `false`), then the global not-enforced URL list would become global enforced list.

Here are a few examples:

Scenario 1:

```
com.sun.am.policy.agents.reverse_the_meaning_of_notenforcedList=
false

com.sun.am.policy.agents.notenforcedList =
http://mycomputer.siroe.com:80/welcome.html
http://mycomputer.siroe.com:80/banner.html
```

In this case, authentication and policies will not be enforced on the two URLs listed in the `notenforcedList`. All other resources will be protected by the agent.

Scenario 2:

```
com.sun.am.policy.agents.reverse_the_meaning_of_notenforcedList=
true

com.sun.am.policy.agents.notenforcedList =
http://mycomputer.siroe.com:80/welcome.html
http://mycomputer.siroe.com:80/banner.html
```

In this case, authentication and policies will be enforced by the agent on the two URLs mentioned in the `notenforcedList`. All other resources will be given access to any user.

Global Not-Enforced IP Address List

The `com.sun.am.policy.agents.notenforced_client_IP_address_list` property is used to specify a list of IP addresses. No authentication is required for the requests coming from these client IP addresses.

In other words, the agent will not protect any web server resources against the accesses coming from the IP addresses in the list.

Enforcing Authentication Only Without Enforcing Policies

The property `com.sun.am.policy.agents.do_sso_only` is used to specify if only authentication is enforced for URLs protected by the agent. If this property is set to `true` (by default it is set to `false`), it indicates that the agent enforces authentication only, without enforcing policies. After an user logs into the Identity Server successfully, the agent will not check for policies related to the user and the accessed URLs.

Forwarding LDAP User Attributes via HTTP Headers

Sun ONE Identity Server Policy Agent has the ability to forward LDAP user attribute values via HTTP headers to end web applications. The LDAP user attribute values come from the server side of Sun ONE Identity Server. The Sun ONE Identity Server Policy Agent behaves like a broker to obtain and relay user attribute values to the destination servlets, CGI scripts, or ASP pages. These applications can in turn use the attribute values to personalize page content.

This feature is configurable through two properties in `AMAgent.properties` file. To turn this feature on and off, use the following `AMAgent.properties` file property:

```
com.sun.am.policy.am.fetchHeaders
```

By default, this property is set to `false`, and the feature turned off. To turn on attribute forwarding, set this property to `true`. To configure the attributes that are to be forwarded in the HTTP headers, use the `AMAgent.properties` file property

```
com.sun.am.policy.am.headerAttributes
```

Below is an example section in the `AMAgent.properties` file which shows how this feature is used:

```
#
# The policy attributes to be added to the HTTP header. The
# specification is of the format
# ldap_attribute_name|http_header_name[,...]. ldap_attribute_name
# is the attribute in data store to be fetched and
# http_header_name is the name of the header to which the value
# needs to be assigned.
#
# NOTE: In most cases, in a destination application where a
# "http_header_name" shows up as a request header, it will be
# prefixed by HTTP_, and all lower case letters will become upper
# case, and any - will become _; For example, "common-name" would
# become "HTTP_COMMON_NAME"
#
com.sun.am.policy.am.headerAttributes=cn|common-name,ou|organiza
tional-unit,o|organization,mail|email,employeenumber|employee-nu
mber,c|country
```

By default, some LDAP user attribute names and HTTP header names are set to sample values.

To find the appropriate LDAP user attribute names, check the following XML file on the machine where the Sun ONE Identity Server server is installed:

SIIS_Install_Dir/SUNWam/config/xml/amUser.xml

The attributes in this file could be either Sun ONE Identity Server User attributes or Sun ONE Identity Server Dynamic attributes. For explanation of these two types of user attributes, refer *Sun ONE Identity Server Administration Guide*.

The attribute and HTTP header names that need to be forwarded must be determined by the end-user applications on the web server that the agent is protecting—after all, these applications are the consumers of the forwarded header values (the forwarded information is used for the customization and personalization of web pages).

NOTE This feature is not available for the Sun ONE Web Proxy Server Agent.

The AMAgent.properties File

The `AMAgent.properties` file stores configuration parameters used by the policy agent. From time to time, you may need to make changes to the default parameters in this file. For example, when you want to specify a different failover web server for running Sun ONE Identity Server.

The `AMAgent.properties` file includes information for the following configurations:

- debugging
- policy agent
- FQDN map
- Sun ONE Identity Server services
- service and agent deployment descriptors
- session failover

The `AMAgent.properties` file also contains configuration information on advanced features, such as forwarding LDAP user attributes through HTTP headers and POST data preservation. The `AMAgent.properties` file has comments before each property; refer to the file for more details.

Table 1-1 provides the default location for `AMAgent.properties` on the various supported servers.

Table 1-1 Locating `AMAgent.properties` on Different Platforms

Server	Location
All Supported UNIX web servers	<div>/etc/opt/SUNWam/agents/<i>WebServer</i>/config/_PathInstanceName/</div> <div>Here, <i>WebServer</i> can be:</div> <div><ul style="list-style-type: none">• es6• es4• proxy• apache• ibmhttp• domino</div>
Sun ONE Web Server 6.0 Windows 2000	<div>\Agent_Install_Dir\Identity_Server\Agents\2.0\es6\config_PathIn</div> <div>stanceName\</div>

Table 1-1 Locating `AMAgent.properties` on Different Platforms

Server	Location
Microsoft IIS 5.0 Windows 2000	<code>\Agent_Install_Dir\Identity_Server\Agents\iis\config_PathInstanceName\</code>
Lotus Domino 5.0.10 Windows 2000	<code>\Agent_Install_Dir\domino\config_PathInstanceName\</code>
Microsoft IIS 4.0 Windows NT	<code>\Agent_Install_Dir\Identity_Server\Agents\iis\config_PathInstanceName\</code>
Apache 1.3.26 Red Hat Linux 7.2	<code>/etc/opt/agents/apache/config/_PathInstanceName/</code>
Apache 1.3.26 Solaris 8 and 9	<code>/etc/opt/SUNWam/agents/apache/config/_PathInstanceName/</code>

Changing the `AMAgent.properties` file can have serious and far-reaching effects. Remember that you can safely change many of the properties in this file by simply re-installing the agent. However, if you must make manual changes, keep the following in mind:

- Make a backup copy of this file before you make changes.
- Trailing spaces are significant; use them judiciously.
- Use forward slash (/) to separate directories, not backlash (\). This holds true even on Windows systems.
- Spaces in the Windows file names are allowed.

NOTE	If you do make changes to the <code>AMAgent.properties</code> file, you must restart the web server to make your changes take effect.
-------------	---

Setting the Fully Qualified Domain Name

To ensure appropriate user experience, it is necessary to enforce that the users use valid URLs to access resources protected by the agent. The configuration property `com.sun.am.policy.agents.fqdnDefault` provides the necessary information needed by the agent to identify if the user is using a valid URL to access the protected resource. If the agent determines that the incoming request does not have

a valid hostname in the URL, it redirects the user to the corresponding URL with a valid hostname. The difference between the redirect URL and the URL originally used by the user is only the hostname, which is changed by the agent to a fully qualified domain name (FQDN) as per the value specified in this property.

This property is a required configuration property without which the Web server may not startup correctly. This property is set during the agent installation and need not be modified unless absolutely necessary to accommodate deployment requirements. Invalid value for this property can result in the Web Server becoming unusable or the resources becoming inaccessible.

The property `com.sun.am.policy.agents.fqdnMap` provides another way by which the agent can resolve malformed access URLs used by the users and take corrective action. The agent gives precedence to the entries defined in this property over the value defined in the `com.sun.am.policy.agents.fqdnDefault` property. If none of the entries in this property match the hostname specified in the user request, the agent uses the value specified for `com.sun.am.policy.agents.fqdnDefault` property.

The `com.sun.am.policy.agents.fqdnMap` property can be used for creating a mapping for more than one hostname. This may be the case when the Web Server protected by this agent is accessible by more than one hostname. However, the use of this feature must be done with caution as it can lead to the Web Server resources becoming inaccessible.

This property can also be used to override the behavior of the agent in cases where necessary. For example, if it is required that no corrective action such as a redirect be used for users who access the Web Server resources using raw IP address, it can be implemented by specifying a map entry such as:

```
com.sun.am.policy.agents.fqdnMap=IP|IP
```

The format for specifying the property `com.sun.am.policy.agents.fqdnMap` is:

```
com.sun.am.policy.agents.fqdnMap =  
[invalid_hostname|valid_hostname][,...]
```

where:

`invalid_hostname` is a possible invalid hostname the user may use such as partial hostname or an IP address.

`valid_hostname` is the corresponding valid hostname which is fully qualified. For example, the following is a possible value specified for hostname `xyz.domain1.com`:

```
com.sun.am.policy.agents.fqdnMap = xyz|xyz.domain1.com,  
xyz.domain1|xyz.domain1.com
```

This value maps `xyz` and `xyz.domain1` to the FQDN `xyz.domain1.com`.

Cookie Reset Feature

This feature enables the Identity Server Policy Agent to reset some cookies in the browser session while redirecting to Identity Server for authentication. Currently, this feature is available only for the Policy Agents for IBM HTTP Server and Lotus Domino.

This feature is configurable through two properties in the `AMAgent.properties` file.

- Enable Cookie Reset

```
com.sun.am.policy.agents.cookie_reset_enabled=true
```

This property must be set to `true`, if this agent needs to reset cookies in the response while redirecting to Identity Server for authentication. By default, this is set to `false`.

- Cookie List

This property gives the comma separated list of cookies, that need to be reset in the response while redirecting to Identity Server for authentication. This property is used only if the Cookie Reset feature is enabled.

Cookie details need to be specified in the following format:

```
name[=value][;Domain=value]
```

For example,

```
com.sun.am.policy.agents.cookie_reset_list=LtpaToken,  
cookie1=value1, cookie2=value2;Domain=siroe.com
```

Configuring CDSSO

The Cross Domain Single Sign-On (CDSSO) feature is configurable through two properties in the file `AMAgent.properties`. To turn this feature on or off, use the following property in `AMAgent.properties`:

```
com.sun.am.policy.agents.cdsso-enabled=true
```

By default, this property is set to `false`, and feature is turned off. To turn on CDSSO, set this property to `true`. Set the URL where CDSSO is installed by specifying the URL in the following property:


```
com.sun.am.policy.agents.loginURL =
http://mycomputer.domain:port/amcdsso/cdsso
```

For more information on configuring CDSSO component, refer *Sun ONE Identity Server Installation Guide*.

Verifying a Successful Installation

After installing a policy agent, it is a good practice to make sure that the agent was installed successfully. There are two things you can check to verify a successful agent installation.

1. Access some web content on the web server where the agent is installed. If the agent is installed correctly, you should see the Sun ONE Identity Server login page. Figure 1-2 is an example of Sun ONE Identity Server login page that uses LDAP authentication.
1. Check the file `AMAgent.properties`. Make sure that each property is set properly.

Figure 1-2 Sun ONE Identity Server Login Page

Sun
Sun ONE Identity Server

This server uses LDAP Authentication

User Name:

Password:

Log In

Sun ONE
Open Net Environment

Copyright 2002 Sun Microsystems, Inc. All rights reserved. Use of this product is subject to license terms.
Federal Acquisitions: Commercial Software -- Government Users Subject to Standard License Terms and
Conditions. Sun, Sun Microsystems, the Sun logo, and iPlanet are trademarks or registered trademarks of
Sun Microsystems, Inc. in the United States and other countries.

Policy Agents for Solaris 8 and 9

Sun ONE Identity Server Policy Agents work in tandem with Sun ONE Identity Server to grant or deny user access to web servers in an enterprise. This chapter explains how to install and configure the policy agents for servers running on the Solaris 8 and Solaris 9 operating systems.

Topics include:

- Before You Begin
- Installation Using the GUI
- Installation Using the Command-Line
- Configuring Agent for Multiple Web Server Instances
- Configuring the Domino DSAPI Filter
- Using Secure Sockets Layer (SSL) with an Agent
- Setting the REMOTE_USER Server Variable
- Validating Client IP Addresses
- POST Data Preservation
- Shared Secret Encryption Utility
- Uninstalling a Policy Agent
- Troubleshooting Solaris Agents

Before You Begin

Be sure that you are familiar with the concepts presented in Chapter 1, “Read This First.” The chapter includes brief but important information on the following topics:

- How Policy Agents Work
- Java Runtime Environment 1.3.1
- The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers
- Configuring Agent for Multiple Web Server Instances on the Same Computer System
- Providing Failover Protection for Sun ONE Identity Server Agents
- Updating the Agent Cache
- Global Not-Enforced URL List
- Global Not-Enforced IP Address List
- Enforcing Authentication Only Without Enforcing Policies
- Forwarding LDAP User Attributes via HTTP Headers
- The AMAgent.properties File
- Setting the Fully Qualified Domain Name
- Configuring CDSSO

Supported Solaris Web Servers

Sun ONE Identity Server Policy Agents support the following servers running on the Solaris 8 operating system:

- Sun ONE Web Server 6.0 SPx
- Sun ONE Web Server 4.1 SP8
- Sun ONE Web Proxy Server 3.6 (in reverse proxy mode)
- Apache 1.3.26
- IBM HTTP Server 1.3.19
- Lotus Domino 5.0.10

Sun ONE Identity Server Policy Agents support the following servers running on the Solaris 9 operating system:

- Sun ONE Web Server 6.0 SPx
- Apache 1.3.26

Patch Cluster for Solaris

When running Apache 1.3.26 Web Server on platforms Solaris 8 and 9, you must ensure that the recommended patches 109234-09 and 113146-01 are installed. You can download these patches from <http://sunsolve.sun.com>

Installation Using the GUI

Use the following instructions for installing agents on the following servers on the Solaris 8 operating system.

To Install a Web Server Policy Agent

You must have root permissions when you run the agent Installation program.

1. Unpack the product binaries using the following commands.

For Sun ONE Web Server 4.1:

```
# gunzip -dc agent_SunOS_es41.tar.gz | tar -xvof -
```

For Sun ONE Web Server 6.0 SPx:

```
# gunzip -dc agent_SunOS_es6.tar.gz | tar -xvof -
```

For Apache 1.3.26 Web Server:

```
# gunzip -dc agent_SunOS_apache.tar.gz | tar -xvof -
```

For IBM HTTP Server 1.3.19:

```
# gunzip -dc agent_SunOS_ibmhttp.tar.gz | tar -xvof -
```

For Lotus Domino 5.0.10

```
gunzip -dc agent_SunOS_domino.tar.gz | tar -xvof -
```

2. Run the `setup` program. You'll find the program in the directory where you untarred the binaries. At the command line, enter the following:

```
# ./setup
```

3. Set your `JAVA_HOME` environment variable to a JDK version 1.3.1_04 or higher. The installation requires that you setup your `JAVA_HOME` variable correctly. However, in case you have incorrectly set the `JAVA_HOME` variable, the setup script will prompt you for supplying the correct `JAVA_HOME` value:

```
Please enter JAVAHOME path to pick up java:
```

4. Type the full path to the directory where JDK is located. The installation program starts with the Welcome page.
5. In the Welcome page, click Next.
6. Read the License Agreement. Click Yes to agree to the license terms.
7. In the Select Installation Directory panel, specify the directory where you would like to install the agent.

Install Sun ONE Identity Server Policy Agent in this directory: Enter the full path to the directory where you want the agent to be installed. The default installation directory is `/opt`.

8. Click Next and provide the following information about the web server the agent will protect:

Host Name: Enter the FQDN of the machine where the web server is installed. For example, `mycomputer.eng.siroe.com`

Web Server Instance Directory: This prompt appears only if you have selected a Sun ONE Web Server agent. Specify the web server instance that this agent will protect. Enter the full path to the directory where the web server instance is located. Example: `/web_server_root/https-mycomputer.siroe.com`

Lotus Domino Data directory: Enter the full path to the directory where the Domino data is located. The default data directory is `/local/notesdata`. This field is available only if you are installing the Policy Agent for Lotus Domino.

Configuration Directory: Specify the directory where the `httpd.conf` file is located. This field appears only when you are installing the agent for Apache server or IBM HTTP server.

Web Server Port: Enter the port number for the web server that will be protected by the agent.

Web Server Protocol: If the web server has been configured for SSL, choose HTTPS; otherwise choose HTTP.

Agent Deployment URI: Enter a directory name. The default Universal Resource Identifier (URI) is `/amagent`.

NOTE The Agent uses value of `com.sun.am.policy.agents.agenturiprefix` property to support some essential functions such as notification and post data preservation. It is important to set a valid URL for this property. Its value should be `http://host.domain:port/agent_deployment_uri` where *host*, *domain* and *port* are FQDN and port number of the web server where the agent is installed and *agent_deployment_uri* is the URI which the web server will look for agent's related HTML pages. Its default value is `amagent`.

SSL Ready: This option is displayed only when you are installing the Apache web server agent. Select this option if the Apache web server you are using has support for SSL. Your apache web server is considered SSL ready if it has support for `mod_ssl` and its sources have been compiled using EAPI rule.

To find out if your Apache web server has been compiled with the EAPI flag, go to the `bin` directory of the Apache web server and type the command:

```
# ./httpd -V
```

You can see various flags that the Apache web server was compiled with. If the flag `-D EAPI` is displayed in this list, it indicates that your Apache Web Server is SSL ready. However, if you do not see this flag, it does not necessarily indicate that the Web Server does not have support for `mod_ssl`.

The supported configuration for Apache web server are:

- a. Apache Web Server without `mod_ssl` support
- b. Apache Web Server with `mod_ssl` and EAPI flag enabled.

NOTE Apache Web Server with `mod_ssl` support and EAPI flag disabled configuration is not supported by Sun ONE Identity Server Policy Agents.

9. When you have entered all the information correctly, click Next.

10. Enter information about the web server that runs Sun ONE Identity Server policy and management. The Policy Agent will connect to this server.

Primary Server Host: Enter the FQDN of the system where the primary web server that runs Sun ONE Identity Server is installed. For example, *myserver.siroe.com*.

Primary Server Port: Enter the port number for the web server that runs Sun ONE Identity Server.

Primary Server Protocol: If the web server that runs Sun ONE Identity Server is SSL-enabled, select HTTPS; otherwise select HTTP.

Primary Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is */amserver*.

Primary Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is */amconsole*.

Failover Server Host: Enter the FQDN for the secondary web server that will run Sun ONE Identity Server if the primary web server becomes unavailable. If no failover host exists, then leave this field blank.

Failover Server Port: Enter the port number of the secondary web server that runs Sun ONE Identity Server. If no failover host exists, then leave this field blank.

Failover Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is */amserver*. If no failover host exists, then leave this field blank.

Failover Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is */amconsole*. If no failover host exists, then leave this field blank.

Agent Identity Server Shared Secret: Enter the password for the Identity Server internal LDAP authentication user (*amldapuser*).

Re-enter Shared secret: Re-enter the password for the Identity Server internal LDAP authentication user.

CDSSO Enabled: Check this box if you want to enable CDSSO.

CDSSO Component URL: Enter the CDSSO Component URL.

11. When all the information is entered correctly, click Next.

12. Review the Installation Summary to be sure that the information you've entered is correct. If you want to make changes, click Back. If all the information is correct, click Next.
13. In the Ready to Install panel, click Install Now.
14. When the installation is complete, you can click Details to view details about the installation, or click Exit to end the Installation program.
15. You must restart your Sun ONE Web Server or Apache web server for the installation to be complete.

NOTE If you are installing the Policy Agent for Lotus Domino 5.0.10, you must configure the Domino DSAPI filter after installation. See the section "Configuring the Domino DSAPI Filter," on page 52 for detailed steps.

Installing a Proxy Server Policy Agent

Use these instructions for installing a Policy Agent on Sun ONE Web Proxy Server 3.6 (in reverse proxy mode) on the Solaris 8 operating system.

To Install a Web Proxy Server Policy Agent

You must have root permissions when you run the agent Installation program.

1. Unpack the product binaries using the following command:

```
# gunzip -dc agent_SunOS_proxy.tar.gz | tar -xvof -
```
2. Run the `setup` program. You will find the program in the directory where you untarred the binaries. At the command line, enter the following:

```
# ./setup
```
3. In the Welcome page, click Next.
4. Read the License Agreement. Click Yes to agree to the license terms.
5. To search for the directory where you would like to install the agent, click Browse. To accept the default, click Next.

6. When prompted, provide the following information about the web proxy server where this agent will be installed:

Host Name: Enter the FQDN of the system where the remote web server is installed. For example, *mycomputer.siroe.com*.

Proxy Server Instance Directory: Enter the full path to the directory where the Sun ONE Web Proxy Server instance is located. For example:

proxy_server_root_dir/proxy-mycomputer-proxy

Proxy Server Port: Enter the port number for the Proxy server instance.

Proxy Server Protocol: If the proxy server has been configured for SSL, choose HTTPS; otherwise choose HTTP.

Agent Deployment URI: Enter a directory name. The default URI is */amagent*.

7. When all the information is entered correctly, click Next.
8. When prompted, provide the following information about the web server that runs Sun ONE Identity Server.

Primary Server Host: Enter the FQDN of the system where the primary web server that runs Sun ONE Identity Server is installed. For example, *myserver.siroe.com*.

Primary Server Port: Enter the port number for the web server that runs Sun ONE Identity Server.

Primary Server Protocol: If the web server that runs Sun ONE Identity Server is SSL-enabled, select HTTPS; otherwise select HTTP.

Primary Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is */amserver*.

Primary Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is */amconsole*.

Failover Server Host: Enter the FQDN for the secondary web server that will run Sun ONE Identity Server if the primary web server becomes unavailable. If no failover host exists, then leave this field blank.

Failover Server Port: Enter the port number of the secondary web server that runs Sun ONE Identity Server. If no failover host exists, then leave this field blank.

Failover Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Failover Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is `/amconsole`.

Agent Identity Server Shared Secret: Enter the password for the Identity Server internal LDAP authentication user.

Re-enter Shared secret: Re-enter the password for the Identity Server internal LDAP authentication user.

CDSSO Enabled: Check this box if you want to enable CDSSO.

CDSSO Component URL: Enter the CDSSO Component URL.

9. When all the information is entered correctly, click Next.
10. Review the Installation Summary to be sure that the information you've entered is correct. If you want to make changes, click Back. If all the information is correct, click Next.
11. In the Ready to Install page, click Install Now.
12. When the installation is complete, you can click Details to view details about the installation, or click Exit to end the Installation program.
13. Restart the Proxy Server.

Installation Using the Command-Line

The command-line version of the Installation program provides you an alternative to the GUI version.

To Install a Web Server Agent Using the Command-Line

1. In the directory where you unpacked the binaries, at the command line, enter the following:

```
# setup -nodisplay
```

2. Set your `JAVA_HOME` environment variable to a JDK version 1.3.1_04 or higher. The installation requires that you set up your `JAVA_HOME` variable correctly. However, in case you have incorrectly set the `JAVA_HOME` variable, the `setup` script will prompt you for supplying the correct `JAVA_HOME` value:

Please enter JAVAHOME path to pick up java:

3. Type the full path to the directory where JDK is located and press Enter.
4. When prompted, provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement? Enter yes.

Install Sun ONE Identity Server Agent in this directory: Enter the full path to the directory in which you want to install the policy agent.

5. Provide the following information about the Web Server this agent will protect:
 - Host Name
 - Port
 - Web Server Instance Directory (Lotus Domino Data Directory if you are installing the Policy Agent for Lotus Domino.)
 - Web Server Protocol
 - Agent Deployment URI
 - SSL Ready (only for Apache agent)

For more information on each of these items, see “To Install a Web Server Policy Agent.”

6. Provide the following information about the web server that runs Sun ONE Identity Server:
 - Primary Server Host
 - Primary Server Port
 - Primary Server Protocol
 - Primary Server Deployment URI
 - Primary Console Deployment URI
 - Failover Server Host
 - Failover Server Port

- Failover Server Deployment URI
- Failover Console Deployment URI
- Agent-Identity Server Shared secret
- Re-enter Shared secret
- CDSSO Enabled
- CDSSO Component URL

For more information on each of these items, see “To Install a Web Server Policy Agent.”

7. The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation
```

When prompted, **What would you like to do?**, enter 1 to start the installation.

8. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Installed	Available
2. Done		

To see log information, enter 1. To exit the Installation program, enter 2.

To Install a Web Proxy Server Agent Using the Command-Line

1. In the directory where you unpacked the binaries, at the command line, enter the following:

```
# setup -nodisplay
```

2. When prompted, provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement? Enter yes.

Install Sun ONE Identity Server Agent in this directory: Enter the full path to the directory in which you want to install the policy agent.

3. Provide the following information about the Web Server this agent will protect:

- Host Name
- Proxy Server Instance Directory
- Proxy Server Port
- Proxy Server Protocol
- Agent Deployment URI

For more information on each of these items, see “To Install a Web Server Policy Agent.”

4. Provide the following information about the proxy web server that runs Sun ONE Identity Server:

- Primary Server Host
- Primary Server Port
- Primary Server Protocol
- Primary Server Deployment URI
- Primary Console Deployment URI
- Failover Server Host
- Failover Server Port
- Failover Server Deployment URI
- Failover Console Deployment URI
- Agent-Identity Server Shared secret
- Re-enter Shared secret
- CDSSO Enabled

- CDSSO Component URL

For more information on each of these items, see “Installing a Proxy Server Policy Agent.”

5. The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation
```

When prompted, **What would you like to do?**, enter 1 to start the installation.

6. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Installed	Available
2. Done		

To see log information, enter 1. To exit the Installation program, enter 2.

Configuring Agent for Multiple Web Server Instances

To configure an Agent for multiple web server instances on a single computer, use the GUI or command-line version of the Agent Installation program to install the first agent. After the first agent is installed, you can then install successive agents using the `config` script. This script must be run from the command line as described in the next section.

In this release, you cannot install more than one type of agents on the same machine. For example, you cannot install an Apache agent and a Sun ONE Web Server agent on the same machine.

To Configure Agent for Multiple Web Server Instances on the Same Computer System

Once you have installed an agent on a system, you can install successive agents on that system using a script that is copied into the system during the agent installation. The two scripts, `config` and `unconfig`, are located in the following directory:

Agent_Install_Dir/SUNWam/agents/*WS_TYPE*/bin

WS_TYPE can be `es6`, `es4`, `proxy`, `apache` or `domino` depending on which web server the agent is protecting.

To install additional agents on a system after the original agent has been installed, run the `config` script from the `bin` directory using the following command:

```
# ./config
```

Follow the prompts to install the additional agents. For information on each of the prompts, see “To Install a Web Server Policy Agent.” In general, information needs to be entered for both the protected web server instance and the Sun ONE Identity Server server(s). The following text shows an example.

If you are installing the Lotus Domino Agent, the following example will show Lotus Domino Data Directory in the place of Web Server Instance Directory.

```
# ./config
Enter the Web Server Instance Directory:
[/web_server_root/https-server_instance]
Enter the Local Hostname: [mycomputer.siroe.com]
Enter the Agent Web Server Port: [80]
Select Agent Web Server Protocol: [1] http [2] https-->[1]
Enter the Agent Deployment URI: [/amagent]
Select Identity Server Protocol: [1] http [2] https --> [1]
Enter the Identity Server Hostname: [mycomputer.siroe.com]
Enter the Identity Server Port: [58080]
Enter the Identity Server Deployment URI [/amserver]
Enter the Identity Server's Console Deployment URI [/amconsole]
Enter the Identity Server's Console Deployment URI [/amconsole]
Select Failover Identity Server Protocol: [1] http [2] https [3]
no failover --> []
Enter the Failover Identity Server Hostname:
[]mycomputer.siroe.com
Enter the Failover Identity Server Port: []
Enter the Identity Server Deployment URI [/amserver]
Enter the Identity Server's Console Deployment URI [/amconsole]
Enter Agent-Identity Server shared secret:
Re-enter Agent-Identity Server shared secret:
```



```
Configuring webserver ... Webserver version: 6.0
Done
```

NOTE The `config` script does not allow you to enter CDSO details. You must configure it manually. For more information, see “Configuring CDSO.”

Using the config Script for Silent Installations

The `config` script can also be used to do silent and non-interactive agent installations. For details on its usage, use the `config -h` command to display details:

```
# ./config -h
Usage: config [ -r response_file | -R | -h ]
       -r specifies a response file.
       -R prints out the response file template.
       -h prints out this message.
```

In order to perform a silent installation, you must supply a *response.file* for each of the agents you want to install. The command `config -R` indicates the fields which you must supply in the *response.file*. This text file needs to be prepared before you begin the silent installation.

```
./config -R
Response file contains:
AGENT_PROTOCOL          # agent protocol: http|https
AGENT_HOST              # agent hostname
AGENT_PORT              # agent server port
AGENT_DEPLOY_URI        # agent deploy URI
FAILOVER_SERVER_HOST    # failover identity server name
FAILOVER_SERVER_PORT    # failover identity server port
FAILOVER_SERVER_DEPLOY_URI # failover identity server deploy URI
FAILOVER_CONSOLE_DEPLOY_URI # failover identity server console deploy URI
PRIMARY_SERVER_HOST     # primary identity server name
PRIMARY_SERVER_PORT     # primary identity server port
PRIMARY_SERVER_PROTO    # primary identity server protocol: http|https
PRIMARY_SERVER_DEPLOY_URI # primary identity server deploy URI
```

```

PRIMARY_CONSOLE_DEPLOY_URI # primary identity server console deploy URI
SHARED_SECRET              # shared secret between agent and DSAME server
SERVER_INSTANCE            # web server instance directory
NOTIFICATION_ENABLE        # notification enabled
AGENT_URL_CASE_IGNORE      # url comparison case ignore

```

Here is an example for *response.file*, named `response.iws60`.

```

AGENT_PROTOCOL=http
AGENT_HOST=mycomputer.siroe.com
AGENT_PORT=80
AGENT_DEPLOY_URI=/amagent
FAILOVER_SERVER_HOST=failover_computer.siroe.com
FAILOVER_SERVER_PORT=58080
FAILOVER_SERVER_DEPLOY_URI=/amserver
FAILOVER_CONSOLE_DEPLOY_URI=/amconsole
PRIMARY_SERVER_HOST=primary_computer.siroe.com
PRIMARY_SERVER_PORT=58080
PRIMARY_SERVER_PROTO=http
PRIMARY_SERVER_DEPLOY_URI=/amserver
PRIMARY_CONSOLE_DEPLOY_URI=/amconsole
SHARED_SECRET=encrypted_shared_secret
SERVER_INSTANCE=/opt/iws6a/https-mycomputer.siroe.com
NOTIFICATION_ENABLE=true
AGENT_URL_CASE_IGNORE=true

```

Below is an example showing how the `config` script is used in conjunction with the `response.iws60` response file to complete a silent installation.

```

# ./config -r ./response.iws60
Configuring webserver ... Webserver version: 60
done

```

NOTE Be sure to use the `unconfig` script to uninstall any agent that was installed using the `config` script—you cannot use the GUI installation program to uninstall agents that were installed from the command-line. The GUI uninstaller must be executed only after unconfiguring all the existing agents using the command-line `unconfig` script.

Removing an Agent Using the unconfig Script

To remove an agent that was installed from the command line using the `config` script, use the script `unconfig`. The `unconfig` script is located in the following directory:

Agent_Install_Dir/SUNWam/agents/*WS_TYPE*/bin

WS_TYPE can be `es6`, `es4`, `proxy`, `apache` or `domino` depending on which web server the agent is protecting.

Here is an example run of the `unconfig` script.

```
# ./unconfig /web_server_root/https-server_instance
Unconfiguring webserver ...
done.
```

NOTE If you are removing the agent for Lotus Domino 5.0.10, you should specify the path to the Domino data directory instead of `https-server_instance`. Additionally, you should remove the DSAPI filter file for the required Domino server using the Domino Administrative Client and then restart the Domino Webserver.

Using Secure Sockets Layer (SSL) with an Agent

During installation, if you choose the HTTPS protocol, the agent is automatically configured and ready to communicate over SSL.

NOTE Before proceeding with the following steps, ensure that the Web Server is configured for SSL.

You should have a solid understanding of SSL concepts and the security certificates required to enable communication over the HTTPS protocol. See the documentation that comes with your web server. If you're using Sun ONE Web Server, you can access the documentation at:

<http://docs.sun.com/source/816-5682-10/esecurty.htm#1011961>

Configuring the IBM HTTP Server

Use the following instructions to configure the IBM HTTP Server to run in SSL mode.

1. Create a new Key Database using the Key Management Utility (IKEYMAN). For information on creating new key database, see the documentation at:
<http://www-3.ibm.com/software/webservers/httpservers/doc/v1319/9atikeyu.htm#HDRKMU2G>
2. Create a Self-signed certificate using the Key Management Utility (IKEYMAN). For information on creating a self-signed certificate, see the documentation at:
<http://www-3.ibm.com/software/webservers/httpservers/doc/v1319/9atikeyu.htm#HDRKMU4G>
3. Start the Administration Server

```
# /opt/IBMHTTPD/bin/adminctl start
```
4. Setup SSL using the IBM Administration Server. For information on setting up SSL, see the documentation at:
<http://www-3.ibm.com/software/webservers/httpservers/doc/v1319/9atstart.htm#ssl>

Configuring the Domino DSAPI Filter

Use the following procedure to configure DSAPI filter:

1. In Lotus Domino Administrator, choose Administrator Tab > Server > All Server Documents.
2. From the listed servers, select the required server.
3. Click Internet Protocols > HTTP tab.
4. At the DSAPI Filter File Names field, enter
Agent_Install_Dir/Agents/Domino/lib/libamdomino.so
5. Click the Save and Close button to save the changes.
6. Open Domino console and restart the server by entering the following commands:

```
tell http quit  
load http
```

Web or Web Proxy Server Running in SSL Mode

If your web or web proxy server is running in SSL mode, and your agent is in notification mode, you must install the root certificate of your web or web proxy server onto the Sun ONE Identity Server if it is not already installed.

The Agent's Default Trust Behavior

By default, the policy agent installed on a remote web server or proxy server will trust any server certificate presented over SSL by the web server that runs Sun ONE Identity Server; the agent does not check the root Certificate Authority (CA) certificate. If the Web Server that runs Sun ONE Identity Server is SSL-enabled, and you want the policy agent to perform certificate-checking, you must do the following:

1. Disable the agent's default trust behavior.
2. Install a root CA certificate on the remote web server (where the agent is installed). The root CA certificate must be the same one that is installed on the web server that runs Sun ONE Identity Server service.

Disabling the Agent's Default Trust Behavior

The following property exists in the `AMAgent.properties` file, and by default it is set to `true`:

```
com.sun.am.policy.agents.trustServerCerts=true
```

This means that the agent does not perform certificate-checking.

To Disable the Default Behavior

1. Set the following property to `false`:

```
com.sun.am.policy.agents.trustServerCerts=false
```

2. Set the directory Cert DB in the file `AMAgent.properties` as shown in the following examples:

For Apache, set as following:

```
com.sun.am.policy.am.sslCertDir= /etc/apache/cert
```

For IBM HTTP Server, set as following:

```
com.sun.am.policy.am.sslCertDir=/opt/IBMHTTPD/cert
```

For Domino Web Server, set as following:

```
com.sun.am.policy.am.sslCertDir=/opt/domino/cert
```

3. Set the Cert DB Prefix, if required.

In cases where the specified Cert DB directory has multiple certificate databases, the following property must be set to the prefix of the certificate database to be used.

```
com.sun.am.policy.am.certDbPrefix
```

For example, set the property for iWS as this:

```
com.sun.am.policy.am.certDbPrefix=https-host.domain.com.host-
```

Installing the Root CA Certificate on the Remote Web Server

The root CA certificate that you install on the remote web server must be the same one that is installed on the web server that runs Sun ONE Identity Server.

To Install the Root CA Certificate on Sun ONE Web Server

See the instructions for installing a root CA certificate in the documentation that comes with the web server. Generally, you install a root CA certificate through the web server's Administration console.

You can access the documentation for Sun ONE Web Server 6.0 on the Internet at the following URL:

```
http://docs.sun.com/source/816-5682-10/esecurty.htm#1011961
```

The procedure for installing a root CA certificate on Sun ONE Web Server 4.1 SP8 is the same.

To Install the Root CA Certificate on Apache 1.3.26

You can use the `certutil` program to install the root CA certificate on Apache 1.3.26.

1. In C shell, at the command line, enter the following commands (assuming `/etc/apache` is the directory where the apache config file is located):

```
# cd /etc/apache/cert
```

```
# setenv LD_LIBRARY_PATH
/Agent_Install_Dir/SUNWam/agents/apache/lib:/Agent_Install_Dir/SUNWam/agents/lib
```

2. Create the necessary certificate database if you have not already done so.

```
# /Agent_Install_Dir/SUNWam/agents/apache/cert/certutil -N -d .
```

3. Install root CA certificate.

```
# /Agent_Install_Dir/SUNWam/agents/apache/cert/certutil -A -n cert-name
-t "C,C,C" -d cert-dir -i cert-file
```

In the commands above, the variables represent the following:

- o *cert-name* can be any name for this root certificate.
- o *cert-dir* is directory where the certificate-related files are located.
- o *cert-file* is the base-64 encoded root certificate file.

For more information on the `certutil` utility, enter `certutil -H` for online Help.

4. To verify that the certificate is properly installed, at the command line, enter the following:

```
# ./certutil -L -d .
```

Trust database information will be displayed including the name of the root CA certificate you installed. For example:

Certificate Name		Trust Attributes
<i>cert-name</i>		C,C,C
p	Valid peer	
P	Trusted peer (implies c)	
c	Valid CA	
T	Trusted CA to issue client certs (implies c)	
C	Trusted CA to certs(only server certs for ssl) (implies c)	
u	User cert	
w	Send warning	

To Install the Root CA Certificate on Web Proxy Server

You can use the `certutil` program to install the root CA Certificate on Proxy Server.

1. In C shell, at the command line, enter the following commands:

```
# mkdir Proxy_Server_Instance_Dir/cert
# cd Proxy_Server_Instance_Dir/cert
# setenv LD_LIBRARY_PATH
/Agent_Install_Dir/SUNWam/agents/proxy/lib:/Agent_Install_Dir/SUNWam/agents/lib
```

2. Create the necessary certificate database if you have not already done so.
3. Install root CA certificate.

```
# /Agent_Install_Dir/SUNWam/agents/proxy/cert/certutil -N -d .
# /Agent_Install_Dir/SUNWam/agents/proxy/cert/certutil -A -n cert-name
-t "C,C,C" -d cert-dir -i cert-file
```

In the commands above, the variables represent the following:

- *cert-name* can be any name for this root certificate.
- *cert-dir* is the directory where the certificate-related files are located.
- *cert-file* is the base-64 encoded root certificate file.

For more information on the `certutil` utility, enter `certutil -H` for online Help.

To Install the Root CA Certificate on IBM HTTP Server 1.3.19

You can use the `certutil` program to install the root CA certificate on IBM HTTP Server 1.3.19.

1. In C shell, at the command line, enter the following commands:

```
# mkdir server_instance_dir/cert
# setenv LD_LIBRARY_PATH /Agent_Install_Dir/SUNWam/agents/
ibmhttp/lib:/Agent_Install_Dir/SUNWam/agents/lib
```

2. Create the necessary certificate database if you have not already done so.

```
# /Agent_Install_Dir/SUNWam/agents/ibmhttp/cert/certutil -N -d .
```

NOTE This will create the following three files in the directory *server_instance_dir/cert*:

- secmod.db
- key3.db
- cert7.db

Make sure that the UNIX user who the IBM HTTP server runs as (for example nobody), has read permissions on these three files.

3. Install root CA certificate.

```
# /Agent_Install_Dir/SUNWam/agents/ibmhttp/cert/certutil -A -n
cert-name -t "C,C,C" -d cert-dir -i cert-file
```

In the command above, the variables represent the following:

- *cert-name* can be any name for this root certificate.
- *cert-dir* is the directory where the certificate-related files are located.
- *cert-file* is the base-64 encoded root certificate file.

For more information on the `certutil` utility, enter `certutil -H` for online Help.

4. To verify that the certificate is properly installed, at the command line, enter the following:

```
# ./certutil -L -d .
```

Trust database information will display including the name of the root CA certificate you installed. Example:

Certificate Name	Trust Attributes
<i>cert-name</i>	C,C,C
p	Valid peer
P	Trusted peer (implies c)
c	Valid CA
T	Trusted CA to issue client certs (implies c)
C	Trusted CA to certs(only server certs for ssl) (implies c)
u	User cert
w	Send warning

To Install the CA Certificate on Domino Web Server

See the instructions for installing a CA Certificate in the documentation that comes with the web server. Generally, this is done through the web server's Administration console.

1. Go to the following directory:

Agent_Install_Dir/Agents/domino/utils

2. Add the same certificate that is installed on the web server that runs Identity Server services into the existing certificate database. At the command line, enter the following command:

```
certutil -A -n cert-name -t "C,C,C" -d cert-dir -i cert-file
```

3. In the command above, the variables represent the following:

- *cert-name* can be any name for this certificate.
- *cert-dir* is directory where the certificate-related files are located. On Solaris, the location is:

Agent_Install_Dir/Agents/domino/cert

- *cert-file* is the base-64 encoded certificate file.

For more information on certutil, type certutil -H

4. Restart Domino Web Server.

Setting the REMOTE_USER Server Variable

The `REMOTE_USER` server environment variable can be set to a Sun ONE Identity Server authenticated user or anonymous user. By setting this variable to a specific user, the user becomes available to web applications (such as a CGI, servlet, or ASP program). This feature makes it possible to personalize the content of displayed HTML pages to specific users.

To enable the `REMOTE_USER` setting for globally not-enforced URLs as specified in the `AMAgent.properties` file (these are URLs that can be accessed by unauthenticated users), you must set the following property in the `AMAgent.properties` file to `TRUE` (by default, this value is set to `FALSE`):

```
com.sun.am.policy.agents.anonRemoteUserEnabled=TRUE
```

When you set this property value to `TRUE`, the value of `REMOTE_USER` will be set to the value contained in the following property in the `AMAgent.properties` file (by default, this value is set to `anonymous`):

```
com.sun.am.policy.agents.unauthenticatedUser=anonymous
```

NOTE This feature is not available for the Sun ONE Web Proxy Server Agent.

Validating Client IP Addresses

This feature can be used to enhance security by preventing the stealing or *hijacking* of SSOTokens.

The `AMAgent.properties` file contains a property titled `com.sun.am.policy.agents.client_ip_validation_enable`, which by default is set to `false`.

If you set this property value to `true`, client IP address validation will be enabled for each in-coming request that contains an SSO token. If the IP address from which request was generated does not match the IP address issued for the SSO token, the request will be denied. This is essentially the same as enforcing a deny policy.

This feature should not be used, however, if the client browser uses a web proxy or if there is a load-balancing application somewhere between the client browser and the agent-protected web server. In such cases, the IP address appearing in the request will not reflect the real IP address on which the client browser runs.

POST Data Preservation

POST data preservation is supported on both Sun ONE Web Server 6.0 SPx agent and Sun ONE Web Server 4.1 SP8 agent. Users can preserve POST data which are submitted to web servers through html forms before users login to the Identity server. Presumably, the html page containing the form should be in global not enforced list. By default, this feature is switched off.

This feature is configurable through two properties in `AMAgent.properties` file. To turn off this feature, use the following `AMAgent.properties` file property and change the value of the property from true to false:

```
com.sun.am.policy.agents.is_postdatapreserve_enabled = true
com.sun.am.policy.agents.postcacheentrylifetime = 10
```

The second property decides how long any POST data can stay valid in the web server cache. After the specified interval, a reaper thread will wake up and clean up any POST cache entries that have lived beyond the specified life time. The following property helps the administrator to configure this time interval. By default, this property is set to 10 minutes.

NOTE This feature is not available for Sun ONE Web Proxy Server and Apache Agent.

Shared Secret Encryption Utility

The Policy Agent stores the shared secret in the `AMAgent.properties` file. By default, this password is the Identity Server internal LDAP authentication user password. This can be changed on the server side by editing the `AMConfig.Properties` file.

The property `com.sun.am.policy.am.password` in the `AMAgent.properties` file is set with the encrypted shared secret while installing the agent.

To reset or change the shared secret, you can use the following utility and set the value in the property.

1. Go to the following directory:

```
Agent_Install_Dir/bin
```

2. Execute the following script from the command line:

```
crypt_util shared_secret
```

3. Cut and paste the output from Step 2 in the property:

```
com.sun.am.policy.am.password
```

4. Restart the Web Server and try accessing any resource protected by the agent.

Uninstalling a Policy Agent

The following sections provide steps for uninstalling the agent.

NOTE Be sure to use the `unconfig` script to uninstall any agent that was installed using the `config` script—you cannot use the GUI installation program to uninstall agents that were installed from the command-line. The GUI uninstaller must be executed only after unconfiguring all the existing agents using the command-line `unconfig` script.

Before Uninstalling the Policy Agent for Lotus Domino 5.0.10

Before you uninstall the Policy Agent for Lotus Domino 5.0.10, you should perform the following steps on the Lotus Domino Administrator client from a Windows machine.

1. Launch Lotus Domino Administrator.
2. Choose Administrator Tab > Server > All Server Documents.
3. From the listed servers, select the server you want to uninstall.
4. Click Internet Protocols > HTTP tab.
5. Remove the DSAPI filter file name specified for the Lotus Domino agent (*Agent_Install_Dir/Agents/Domino/lib/libamdomino.so*).
6. Click the Save and Close button to save the changes.
7. Open the Domino console and restart the server by entering the following commands:

```
tell http quit
load http
```

Uninstalling Using the GUI

To uninstall an agent, you must run the Uninstallation program. Follow the steps below:

1. In the directory where the agent is installed, at the command line, enter the following command:

```
# ./uninstall_agent
```

2. Click Next on Welcome Panel.
3. Click Uninstall Now on Ready to Uninstall Panel.
4. Click Close after uninstallation is complete.

In the directory where the product binary file is unpacked (and where the installation program is invoked), there is another uninstallation program named `uninstall`. This `uninstall` program can be used to detect and uninstall all agents that were previously installed using the `setup` program; even agents that were installed on a remote machine. Invoke `uninstall` using the following command:

```
# uninstall
```

On the other hand, `uninstall_agent` will uninstall only the agent (or agents) that were previously installed with the `setup` program in the current directory.

Uninstalling Using the Command-Line Interface

1. From the directory where the agent is installed, enter the following command at the command line:

```
# ./uninstall_agent -nodisplay
```

The uninstaller detects the agent that was previously installed using the `setup` program. Enter 1 to uninstall the agent.

The following text is displayed:

```
Ready to Uninstall

1. Uninstall Now
2. Start Over
3. Exit Uninstallation
```

2. When prompted, **What next?** enter 1 to begin uninstallation.

The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Full	Available
2. Done		

To see log information, enter 1. To exit the Installation program, enter 2.

Troubleshooting Solaris Agents

Cannot install Agent after previous installation is removed

The following is an example message that is displayed when you run the Agent installer:

```
"Sun ONE Identity Server Policy Agent 2.0 for Sun ONE Web Server
6.0 SPx is installed. Please refer to installation manual to
configure this agent for another web server instance. Or
uninstall it before installing another agent."
```

Possible Causes:

- You might have an existing installation of Agent.
- You might have a previously installed Agent and did not use Agent's uninstaller to uninstall the Agent
- The installer's `productregistry` file may be corrupted.

Solution:

- Check that you have uninstalled any existing installation of Agent.
- The `productregistry` file may be corrupted if there is no existing installation of Agent. This file is used by the installer to track installed products. It is found in `/var/sadm/install` directory.

NOTE Make a backup copy of this file before you make changes.

Remove the Agent product entry in this file. This entry starts with the following lines:

```
<compid>SUNWamcom
  <compversion>2.0
    <uniquename>SUNWamcom</uniquename>
    <vendor></vendor>
    .....
  </compid>
  <compid>Agent uninstall script
    <compversion>2.0
      <uniquename>Agent uninstall script</uniquename>
      <vendor>Sun Microsystems, Inc.</vendor>
      .....
    </compid>
    <compid>Agent installer resource bundle
      <compversion>2.0
        <uniquename>Agent installer resource
bundle</uniquename>
        <vendor>Sun Microsystems, Inc.</vendor>
        .....
      </compid>
      <compid>Agent Common Core and SDK
        <compversion>2.0
          <uniquename>Agent Common Core and SDK</uniquename>
          <vendor></vendor>
          .....
        </compid>
        <compid>SUNWames6
          <compversion>2.0
            <uniquename>SUNWames6</uniquename>
            <vendor></vendor>
            .....
          </compid>
          <compid>Agent for ...
            <compversion>2.0
              <uniquename>Agent for ...</uniquename>
              <vendor></vendor>
              .....
            </compid>
            <compid>Sun ONE Identity Server Policy Agent
              <compversion>2.0
                <uniquename>Sun ONE Identity Server Policy
Agent</uniquename>
              </compid>
```


The Uninstallation program does not remove entries from the agent's web server if there is another instance of web server configured using the Config script.

You should remove all the instances of the agent using the `unconfig` script before running the Uninstallation program.

Domino Web Server starts with an error message "Unable to load filter".

Ensure that you have set the DSAPI filter correctly. For steps on configuring the Domino DSAPI filter, see the section Configuring the Domino DSAPI Filter.

Domino DSAPI Filter is not functioning properly on the partitioned server.

Possible Cause: The database you have selected while configuring the DSAPI Filter may be wrong.

Solution: Ensure that you have selected the correct database while configuring the DSAPI filter.

Possible Cause: The partitioned database might not have been updated.

Solution: You might have to replicate the database from the Domino Admin Server.

The Agent's performance may suffer when multiple threads are running.

Solution:

On Solaris 8:

1. Apply the latest Solaris 8 cluster patch (Patch Cluster Date Jan/24/03 or later).
2. Download and install patch 111308-03 or later (`showrev -p | grep 111308`) from <http://sunsolve.central.sun.com>.
3. Reboot the machine after applying the patches (this is recommended but not mandatory).

On Solaris 9:

1. Apply the latest Solaris 9 cluster patch (Patch Cluster Date Jan/23/03 or later).
2. Download and install patch 111308-03 or above (`showrev -p | grep 111308`) from <http://sunsolve.central.sun.com>.
3. Reboot the machine after applying the patches (it is recommended but not mandatory).

For Agents running on Sun ONE Web Server 6.0:

1. In your web server instance directory open the start script
2. Before the first `LD_LIBRARY_PATH` line, add the following two lines

```
LD_PRELOAD=libmtmalloc.so.1
```

```
export LD_PRELOAD
```

3. To make sure the patches are installed correctly, try this. It should produce a line of output.

```
showrev -p | grep 111308
```

4. To verify that `libmtmalloc` is being used by the web server, do the following.

```
ps -ef |grep httpd | grep nobody
```

5. Take the pid from the previous step and do

```
pldd pid of the process | grep mtmalloc
```

The output should show `/usr/lib/libmtmalloc.so.1`.

Known Problems

After installing the Solaris 2.8 Patch # 109234-09, Apache Server does not startup correctly or hangs

This can happen if your system does not have `JServ` installed at the time of installing the patch. In order to correct this problem, edit your `httpd.conf` file located under `/etc/apache/` directory and comment the following line:

```
LoadModule jserv_module /usr/apache/libexec/mod_jserv.so and include
/etc/apache/jserv.conf
```

Error message displayed during startup

Apache server displays the following error message during startup after the agent is installed:

```
Syntax error on line 1 of
/etc/opt/SUNWam/agents/apache/config/_usr_local_apache_conf/dsam
e.conf:
```

```
Invalid command 'LoadModule', perhaps mis-spelled or defined by a
module not included in the server configuration
./apachectl start: httpd could not be started
```

This indicates that the Apache server does not have `mod_so` enabled and consequently does not support dynamic shared objects. To enable `mod_so` support, refer Apache Server documentation at <http://httpd.apache.org/>.

Policies not working with Sun ONE Web Proxy Server Agent.

The resource names for setting policies for reverse proxy agent should be the URLs qualified as “URL prefix (from client)” in the Sun ONE Web Proxy Server 3.6 admin console.

Agents are not effective any more after modification of Sun ONE Web Server or Sun ONE Web Proxy Server configuration using the respective admin console.

The changes by agent installation to the server configuration files are overwritten by saving the changes in admin console.

The right procedure when using the admin console should be to load configuration first (from disk file to memory), then make modification, and save the changes (from memory to disk) by clicking the Apply button.

Policy Agents for Windows 2000

Sun ONE Identity Server Policy Agents work in tandem with Sun ONE Identity Server to grant or deny user access to web servers in an enterprise. This chapter explains how policy agents can be configured for various web servers running on the Windows 2000 operating system.

Topics include:

- Before You Begin
- Installation Using the GUI
- Installation Using the Command-Line
- Configuring the Domino DSAPI Filter
- Using Secure Sockets Layer (SSL) with an Agent
- Setting the REMOTE_USER Server Variable
- Validating Client IP Addresses
- POST Data Preservation
- Shared Secret Encryption Utility
- Troubleshooting the Installation

Before You Begin

Be sure that you are familiar with the concepts presented in Chapter 1, “Read This First.” The chapter includes brief but important information on the following topics:

- How Policy Agents Work

- Java Runtime Environment 1.3.1
- The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers
- Configuring Agent for Multiple Web Server Instances on the Same Computer System
- Providing Failover Protection for Sun ONE Identity Server Agents
- Updating the Agent Cache
- Global Not-Enforced URL List
- Global Not-Enforced IP Address List
- Enforcing Authentication Only Without Enforcing Policies
- Forwarding LDAP User Attributes via HTTP Headers
- The AMAgent.properties File
- Setting the Fully Qualified Domain Name
- Configuring CDSSO

Supported Windows Web Servers

The Sun ONE Identity Server Policy Agents support the following web servers on the Windows 2000 server operating system:

- Microsoft IIS 5.0
- Sun ONE Web Server 6.0 SPx
- Lotus Domino 5.0.10

Installing the Policy Agent for Microsoft IIS

The IIS agent enforces policy on URL access for Microsoft's Internet Information Services (IIS) web server. The agent is an IIS ISAPI filter installed at the IIS web service level that will enforce policy on all IIS web sites. Technical considerations prevent the agent from being installed at the web site level.

Prior to installation, be sure that the entry for the system where the agent will be installed has a domain name set. If the web server that runs Sun ONE Identity Server 6.0 is running on a separate system, make sure the server is also in the DNS query list.

Installation Using the GUI

Use the following instructions for installing agents on the Microsoft Windows 2000 operating system:

To Install the Policy Agent

You must have administrator privileges to run the installation program.

1. Unzip the product binaries.

For Microsoft IIS

- o Unzip `agent_WINNT_iis.zip`

For Sun ONE Web Server 6.0 SPx

- o Unzip `agent_WINNT_es6.zip`

For Lotus Domino 5.0.10

- o Unzip `agent_WINNT_domino.zip`

2. Run the Installation program by double-clicking `setup.exe`.
3. In the Welcome window, click Next.
4. Read the License Agreement. Click Yes to accept the license agreement.
To search for the directory where you would like to install the agent, click Browse. To accept the default, click Next.
5. Enter the information about the web server where this agent will be installed:

Host Name: Enter the FQDN of the system where the agent web server is installed. For example, `mycomputer.siroe.com`.

IIS Document Root: Enter the document root directory. This directory needs to be accessible by the web server root `w3svc`. This field is available only if you are installing the Policy Agent for Microsoft IIS.

Web Server Instance Directory: Enter the full path to the directory where the Sun ONE Web Server instance is located. This is the web server instance that the agent will protect. For example,

/web_server_root/https-mycomputer.siroe.com

This field is available only if you are installing the Policy Agent for Sun ONE Web Server.

Lotus Domino Data directory: Enter the full path to the directory where the Domino data is located. The default path is *c:\Lotus\Domino*. This field is applicable only if you are installing the Policy Agent for Lotus Domino 5.0.10.

Server Port: Enter the port number for the web server that will be protected by the agent.

Server Protocol: If your web server has been configured for SSL, then select HTTPS; otherwise select HTTP.

Agent Deployment URI: Enter a directory name. The default Universal Resource Identifier (URI) is */amagent*.

NOTE	Agent uses the value of <i>com.sun.am.policy.agents.agenturiprefix</i> property to support some essential functions such as notification and post data preservation. It is important to set a valid URL for this property. Its default value is: <i>http://host.domain:port/agent_deployment_uri</i> where <i>host</i> , <i>domain</i> and <i>port</i> are the FQDN and port number of the server where agent is installed. <i>agent_deployment_uri</i> is the URI using which the web server will access the agent's HTML pages. Its default value is <i>amagent</i> .
-------------	---

When all the information is entered correctly, click Next.

6. Provide the following information about the web server that runs Sun ONE Identity Server:

Primary Server Host: Enter the FQDN of the system where the primary web server that runs Sun ONE Identity Server is installed. For example, *myserver.siroe.com*.

Primary Server Port: Enter the port number for the web server that runs Sun ONE Identity Server.

Primary Server Protocol: If the web server that runs Sun ONE Identity Server is SSL-enabled, select HTTPS; otherwise select HTTP.

Primary Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Primary Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is `/amconsole`.

Failover Server Host: Enter the FQDN for the secondary web server that will run Sun ONE Identity Server if the primary web server becomes unavailable. If no failover host exists, then leave this field blank.

Failover Server Port: Enter the port number of the secondary web server that runs Sun ONE Identity Server. If no failover host exists, then leave this field blank.

Failover Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`. If no failover host exists, then leave this field blank.

Agent Identity Server Shared Secret: Enter the password for the Identity Server internal LDAP authentication user.

Re-enter Shared secret: Re-enter the password for the Identity Server internal LDAP authentication user.

CDSSO Enabled: Check this box if you want to enable the CDSSO feature.

CDSSO Component URL: Enter the CDSSO Component URL.

7. If all the information entered is correct, click Next.
8. Review the Installation Summary to be sure that the information you've entered is correct. If you want to make changes, click Back. If all the information is correct, click Next.
9. In the Ready to Install page, click Install Now.
10. When the installation is complete, you can click Details to view details about the installation, or click Close to end the Installation program.

If you are installing the Policy Agent for Lotus Domino 5.0.10, you should configure the Domino DSAPI filter. For steps to do this, see "Configuring the Domino DSAPI Filter," on page 76.

The installation modifies the system path by appending to it the location of the Agent libraries. In order for the change to take effect and for the Agent to work properly, you must reboot your computer.

NOTE If the IIS 5.0 or the Lotus Domino 5.0.10 Policy agent was previously installed and uninstalled on your machine, you do not need to reboot if you are installing the same agent in the same directory.

Installation Using the Command-Line

The command-line version of the Installation program provides you an alternative to the GUI-based installation.

Installing an Agent Using the Command-Line

1. In the directory where you unzipped the binaries, at the command line, enter the following command:

```
# setup.bat -nodisplay
```

2. When prompted, provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement?

Install Sun ONE Identity Server Policy Agent in this directory: Specify the directory where you want the agent to be installed. To accept the default directory that is displayed in brackets, press Enter. Otherwise, enter the full path.

3. When prompted, provide the following information about the web server instance this Agent will protect:
 - Host Name
 - IIS Document Root, Web Server Instance Directory, or Lotus Domino Data Directory depending on the agent you are installing.
 - Server Port
 - Server Protocol

- Agent Deployment URI

For details on these items, see “Installation Using the GUI.”

4. When prompted, provide the following information about the web server that runs Sun ONE Identity Server Services:
 - Primary Server Host
 - Primary Server Port
 - Primary Server Protocol
 - Primary Server Deployment URI
 - Primary Console Deployment URI
 - Failover Server Host
 - Failover Server Port
 - Failover Server Deployment URI
 - Secondary Console Deployment URI
 - Agent-Identity Server Shared Secret
 - Re-enter Shared secret
 - CDSSO feature enabled
 - CDSSO component URL

For details on these items, see “Installation Using the GUI.”

5. When displayed, review the summary of installation information you’ve specified. Press Enter to continue, or enter exclamation mark (!) to exit the program.

The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation

What would you like to do
```

6. When prompted, **What would you like to do?**, enter 1 to start the installation.

The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Installed	Available
2. Done		

7. To see log information, enter 1. To exit the Installation program, enter 2.

The installation modifies the system path by appending to it the location of the Agent libraries. In order for the change to take effect and for the Agent to work properly, you must reboot your computer.

NOTE If the IIS 5.0 or the Lotus Domino 5.0.10 Policy agent was previously installed and uninstalled on your machine, you do not need to reboot if you are installing the same agent in the same directory.

If you are installing the Policy Agent for Lotus Domino 5.0.10, you should configure the Domino DSAPI filter. For steps to do this, see the section “Configuring the Domino DSAPI Filter.”

Configuring the Domino DSAPI Filter

Use the following procedure to configure DSAPI filter.

1. In Lotus Domino Administrator, choose Administrator Tab > Server > All Server Documents.
2. From the listed servers, select the server you want to configure.
3. Click Internet Protocols > HTTP tab.
4. At the DSAPI Filter File names field, enter *Agent_Install_Dir\domino\bin\amdomino.dll*.
5. Save the changes and close the window.
6. Open Domino console and restart the server by entering the following commands:

```
tell http quit
load http
```

Configuring Domino DSAPI Filter for Multiple Server Partitions

If you are configuring Domino DSAPI Filter for multiple server partitions, you must:

- Use the same `AMAgent.properties` file for all the supported partitions.
- Configure the filter for each of the server partitions you want to support.

You can configure the filter for the different partitions by performing the following tasks:

1. In Lotus Domino Administrator, choose Administrator Tab > Server > All Server Documents.
2. From the listed servers, select the required server.
3. Now go to Internet Protocols > HTTP.
4. At the DSAPI Filter File names field, enter *Agent_Install_Dir\domino\bin\amdomino.dll*.
5. Save the changes and close the window.
6. Open Domino console and restart the server by entering the following commands:

```
tell http quit
load http
```

Using Secure Sockets Layer (SSL) with an Agent

During installation, if you choose the HTTPS protocol, the agent is automatically configured and ready to communicate over SSL.

NOTE

Before proceeding with the following steps, ensure that the Web Server is configured for SSL.

You should have a solid understanding of SSL concepts and the security certificates required to enable communication over the HTTPS protocol. See the documentation that comes with your web server. If you're using Sun ONE Web Server, you can access the following documentation on the Internet:

<http://docs.sun.com/source/816-5682-10/esecurty.htm#1011961>

The Agent's Default Trust Behavior

By default, the policy agent installed on a remote Sun ONE Web Server 6.0 or Microsoft IIS 5.0 will trust any server certificate presented over SSL by the web server that runs Sun ONE Identity Server; the agent does not check the root Certificate Authority (CA) certificate. If the web server that runs Sun ONE Identity Server is SSL-enabled, and you want the policy agent to perform certificate-checking, you must do two things:

1. Disable the agent's default trust behavior.
2. Install a root CA certificate on the remote web server where the agent is installed. The root CA certificate must be the same one that is installed on the web server that runs Sun ONE Identity Server.

Disabling the Agent's Default Trust Behavior

The following property exists in the `AMAgent.properties` file, and by default it is set to `true`:

```
com.sun.am.policy.agents.trustServerCerts=true
```

This means that the agent does not perform certificate checking.

To Disable the Default Behavior

The following property must be set to `false`:

```
com.sun.am.policy.agents.trustServerCerts=false
```

Installing the Root CA Certificate on the Remote Web Server

The root CA certificate that you install on the remote web server must be the same one that is installed on the web server that runs Sun ONE Identity Server.

To Install the Root CA Certificate on Sun ONE Web Server

See the instructions for installing a root CA Certificate in the documentation that comes with the web server. Generally, this is done through the web server's Administration console. Access the documentation for Sun ONE Web Server 6.0 on the Internet at the following URL:

<http://docs.sun.com/source/816-5682-10/esecurty.htm#1011961>

To Install the Root CA Certificate on Microsoft IIS

1. Go to the following directory:

Agent_Install_Dir\iis\cert

2. Add the same root certificate that is installed on the web server that runs Sun ONE Identity Server into the existing certificate database. At the command line, enter the following command:

```
\Agent_Install_Dir\bin\certutil -A -n cert-name -t "C,C,C" -d cert-dir -i cert-file
```

using the following variables:

- *cert-name* can be any name for this root certificate.
- *cert-dir* is directory where the certificate-related files are located. On Windows the location is:

Agent_Install_Dir\bin

- *cert-file* is the base-64 encoded root certificate file.

- For more information on `certutil`, type `certutil -H`

To verify that the root certificate was installed properly in the certificate database, enter the following command:

```
Agent_Install_Dir\bin\certutil -L -d .
```

You should see the root certificate added and listed in the output of the command.

3. Restart IIS.

To Install the CA Certificate on Domino Web Server

The CA certificate that you install on the Domino Web server must be the same one that is installed on the web server that runs Identity Server services.

See the instructions for installing a CA Certificate in the documentation that comes with the web server. Generally, this is done through the web server's Administration console.

1. Go to the following directory:

```
Agent_Install_Dir\Agents\domino\utils
```

2. Add the same certificate that is installed on the web server that runs Identity Server services into the existing certificate database. At the command line, enter the following command:

```
certutil -A -n cert-name -t "C,C,C" -d cert-dir -i cert-file
```

using the following variables:

- *cert-name* can be any name for this certificate.
- *cert-dir* is directory where the certificate-related files are located. On Windows the locations is:

```
Agent_Install_Dir\Agents\domino\cert
```

- *cert-file* is the base-64 encoded certificate file.

For more information on `certutil`, type `certutil -H`

3. Restart Domino Web Server.

Setting the REMOTE_USER Server Variable

The `REMOTE_USER` server environment variable can be set to a Sun ONE Identity Server authenticated user or an anonymous user. By setting this variable to a specific user, the user becomes available to web applications (such as a CGI, servlet, or an ASP program). This feature makes it possible to personalize the content of displayed HTML pages to specific users.

Performing these steps will set `REMOTE_USER` for allowed URLs.

To enable the `REMOTE_USER` setting for globally not-enforced URLs as specified in the `AMAgent.properties` file (these are URLs that can be accessed by unauthenticated users), you must set the following property in the `AMAgent.properties` file to `TRUE` (by default, this value is set to `FALSE`):

```
com.sun.am.policy.agents.anonRemoteUserEnabled=TRUE
```

When you set this property value to `TRUE`, the value of `REMOTE_USER` will be set to the value contained in the following property in the `AMAgent.properties` file (by default, this value is set to `anonymous`):

```
com.sun.am.policy.agents.unauthenticatedUser=anonymous
```

To enable the `REMOTE_USER` feature for an IIS 5.0 agent, perform the following steps:

1. From the Windows Start menu, select Programs > Administrative Tools > Internet Services Manager.
This will launch the Internet Information Services console.
2. On the web site that you want the Sun ONE Identity Server agent to protect, select Properties.
3. Select the Directory Security tab.
4. In the Anonymous Access and Authentication Control section, click Edit.
5. In the dialog that displays, select Anonymous Access and Basic Authentication, then deselect Integrated Windows Authentication.

Validating Client IP Addresses

This feature can be used to enhance security by preventing the stealing or *hijacking* of SSO tokens.

The `AMAgent.properties` file contains a property titled `com.sun.am.policy.agents.client_ip_validation_enable`, which by default, is set to `false`.

If you set this property value to `true`, client IP address validation will be enabled for each in-coming request that contains an SSO token. If the IP address from which request was generated does not match the IP address issued for the SSO token, the request will be denied. This is essentially the same as enforcing a deny policy.

This feature should not be used, however, if the client browser uses a web proxy or if there is a load-balancing application somewhere between the client browser and the agent-protected web server. In such cases, the IP address appearing in the request will not reflect the real IP address on which the client browser runs.

POST Data Preservation

POST data preservation is supported on Sun ONE Web Server 6.0 SPx agent. Users can preserve POST data which are submitted to web servers through html forms before users login to the Identity server. Presumably the html page containing the form should be in global not enforced list. By default, this feature is turned off.

This feature is configurable through two properties in `AMAgent.properties` file. To turn off this feature, use the following `AMAgent.properties` file property and change the value of the property from `true` to `false`:

```
com.sun.am.policy.agents.is_postdatapreserve_enabled=true
```

The second property decides how long any POST data can stay valid in the web server cache. After the specified interval, a reaper thread will wake up and clean up any POST cache entries that have lived beyond the specified life time. The following property helps the administrator to configure this time interval. By default this property is set to 10 minutes.

```
com.sun.am.policy.agents.postcacheentrylifetime=10
```

NOTE	This feature is not available for the IIS 5.0 agent on win2k.
-------------	---

Shared Secret Encryption Utility

The Policy Agent stores the shared secret in the `AMAgent.properties` file. By default, this password is the Identity Server internal LDAP authentication user password. This can be changed on the server side by editing the `AMConfig.Properties` file.

The property `com.sun.am.policy.am.password` in the `AMAgent.properties` file is set with the encrypted shared secret while installing the agent.

To reset or change the shared secret, you can use the following utility and set the value in the property.

1. Go to the following directory:
`Agent_Install_Dir\bin`
2. Execute the following script from the command line
`cryptit shared_secret`
3. Cut and paste the output from Step 2 in the property:
`com.sun.am.policy.am.password`
4. Restart the Web Server and try accessing any resource protected by the agent.

Uninstalling and Disabling Policy Agent

When you no longer require the policy agent, you can uninstall it or disable it.

Uninstalling a Policy Agent

1. From the Windows Start menu, choose Settings > Control Panel.
2. In the Control Panel, open Add / Remove Programs.
3. In the Add/Remove Programs window, choose Sun ONE Identity Server Policy Agent.
4. Click Change/Remove.
5. Click Next on Welcome panel.
6. Click Uninstall Now.
7. Click Exit after uninstallation is complete.

Disabling a Policy Agent Installed on Microsoft IIS

Use the following steps to disable an agent installed on Microsoft IIS:

1. Launch Internet Services Manager.
 - From the Start menu, choose Programs > Administrative Tools > Internet Services Manager.
2. Check the filter status.
 - a. Open properties for the host computer in the tree pane of the Internet Services Manager window which is titled “Internet Information Services.”
 - b. The host computer name should appear in the tree underneath the Internet Information Services root.
 - c. Click Edit in the Master Properties section of the Internet Information Services tab.
 - d. Select the ISAPI Filters tab in the WWW Service Master Properties dialog that appears.
 - e. Highlight the filter named “Sun ONE Identity Server Agent.”

You can click Edit to view the filter name and executable path. You’ll need this information when you want to re-enable the agent. Click Cancel to return to the program.
 - f. Click Remove.
 - g. Click Apply and exit from the WWW Service Master Properties dialog.
 - h. Restart Microsoft IIS.

Uninstalling the Policy Agent for Lotus Domino 5.0.10

Before you uninstall the Policy Agent for Lotus Domino 5.0.10, you should perform the following steps on the Lotus Domino Administrator client from a Windows machine.

1. Launch Lotus Domino Administrator.
2. Choose Administrator Tab > Server > All Server Documents.

3. From the listed servers, select the server you want to uninstall.
4. Click Internet Protocols > HTTP tab.
5. Remove the DSAPI filter file name specified for the agent and leave this field blank.
6. Click the Save and Close button to save the changes.
7. Open Domino console and restart the server by entering the following commands:


```
tell http quit
load http
```
8. From the Start Menu, choose Settings > Control Panel.
9. In the Control Panel, double-click Add / Remove Programs.
10. In the Add/Remove Programs window, choose Sun ONE Identity Server Policy Agent and click on Change/Remove.
11. In the Welcome Panel, click Next.
12. In the Ready to Uninstall Panel, click Uninstall Now.
13. Click Exit after uninstallation is complete.

Uninstalling an Agent Using the Command-Line

1. In the *Agent_Install_Dir* directory, at the command line, enter the following command:


```
java uninstall_Sun_ONE_Identity_Server_Policy_Agent -nodisplay
```
2. The following text is displayed:

```
1. Uninstall Now
2. Start Over
3. Exit Uninstallation
What would you like to do?
```

When prompted, **What would you like to do?**, enter 1 to start the installation.

3. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Full	Available
2. Done		

To see log information, enter 1. To exit the uninstallation program, enter 2.

4. When the uninstallation is completed, you must reboot the system.

If you want to see more details of the uninstallation, a log file is written in the following location:

`%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall*`

Troubleshooting the Installation

IIS Policy Agent

If you are experiencing problems with your installation try the following:

- Check the installation log file for errors:
`%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall.nnnn`
- Re-install by uninstalling and then installing.
- Verify agent loading in IIS:
 - a. Launch Internet Services Manager.
 - b. From the Start menu, choose Programs > Administrative Tools > Internet Services Manager.
 - c. Open the properties for the host computer in the Tree Pane of the Internet Services Manager window that is titled Internet Information Services.
 - d. The host computer name should appear in the tree underneath the Internet Information Services root.
 - e. Click Edit in the Master Properties section of the Internet Information Services tab.

f. Select the ISAPI Filters tab in the WWW Service Master Properties dialog that appears.

g. Look for the filter name “Sun ONE Identity Server agent.”

If the Filter name “Sun ONE Identity Server Agent” does not appear at all, then check that the installer was run, and look for any errors during installation. The install log is located at:

`%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall.nnnn`

A green arrow pointing up in the Status column to the right of the “Sun ONE Identity Server Agent” indicates the agent loaded successfully into IIS. A red arrow pointing down indicates that the filter failed to load. The most likely cause of the filter not loading successfully (red arrow) is that it cannot locate the required dll files.

h. Check your system path to ensure that the following directory is present:

Agent_Install_Dir\bin

i. If the filter did not load successfully check the following:

- Check the path of the Agent DLL by clicking “Sun ONE Identity Server Agent” and then Edit. Ensure that the path in the text box labeled Executable is valid.
- The agent also needs several DLL files. Check that the following exist in the directory `Agents\bin`:

`amsdk.dll`

`ames6.dll`

`libnspr4.dll`

`libplc4.dll`

`libplds4.dll`

`libxml2.dll`

`nss3.dll`

`ssl3.dll`

j. If the libraries are in your system path try rebooting the system.

- IIS logs filter loading errors in the System Event Log. To check the event log:

a. From the Start menu, choose Programs > Administrative Tools > Event Viewer.

- b. Select the System Log.
- c. Check for Error messages with Source W3SVC.
- If the agent loads but returns HTTP 500 Internal Server Error for all URL requests to the IIS web server.

This indicates that the agent has loaded but did not properly initialize. Returning HTTP 500 Internal Server Error for all HTTP requests is a fail-safe to protect URL resources when the Agent cannot initialize. The most likely cause is a Sun ONE Identity Server agent or server misconfiguration or unavailability.

- Check the agent debug log.

The log is located by default at the *Agent_Install_Dir* directory. This is the best source of debug information for resolving initialization and agent operation issues. The log file directory is specified by the property:

`com.sun.am.policy.am.logFile` in the `AMAgent.properties` file located in the directory:

Agent_Install_Dir\iis\config_PathInstanceName

The property `com.sun.am.policy.am.loglevels` controls the verbosity of the log information. Set the logging level for the specified logging categories.

The format of the values is:

*ModuleName[: Level] , ModuleName[: Level]] **

The currently used module names are AuthService, NamingService, PolicyService, SessionService, PolicyEngine, ServiceEngine, Notification, PolicyAgent, RemoteLog and all. If the level is omitted, then the logging module will be created with the default logging level, which is the logging level associated with the 'all' module.

The all module can be used to set the logging level for all modules. This will also establish the default level for all subsequently created modules. The meaning of the 'Level' value is described below:

- 0 = Disable logging from specified module
- 1 = Log error messages
- 2 = Log warning and error messages
- 3 = Log info, warning, and error messages
- 4 = Log debug, info, warning, and error messages

5 = Like level 4, but with even more debugging messages.

- Check that the agent can locate the `AMAgent.properties` configuration file.

The agent uses the registry key `HKEY_LOCAL_MACHINE\Software\Sun Microsystems\Identity Server IIS Agent` to locate the `AMAgent.properties` file. The `AMAgent.properties` file is located at:

Agent_Install_Dir\iis\config_PathInstanceName

- The agent uses the Application Event Log to log errors that occur before the debug log file specified in `AMAgent.properties` is started.
 - a. From the Start menu, choose Programs > Administrative Tools > Event Viewer.
 - b. Select the Application Log.
 - c. Check for Error messages with Source Sun ONE Identity Server IIS Agent.

Cannot install Agent after previous installation is removed

The following is an example message displayed when you run the Agent installer:

"Sun ONE Identity Server Policy Agent 2.0 for Sun ONE Web Server 6.0 SPx is installed. Please refer to installation manual to configure this agent for another web server instance. Or uninstall it before installing another agent."

Possible Causes:

- You might have an existing installation of Agent;
- You might have a previously installed Agent and did not use Agent's uninstallation program to uninstall the Agent
- The installer's `productregistry` file may be corrupted.

Solution:

- Check that you have uninstalled any existing installation of Agent.
- The `productregistry` file may be corrupted if there is no existing installation of Agent. This file is used by the installer to track installed products. It is found in `C:\WINNT\system32` directory.

NOTE Make a backup copy of this file before you make changes.

Remove the Agent product entry in this file. This entry starts with the following lines:

```
<compid>SUNWamcom
    <compversion>2.0
    <uniquename>SUNWamcom</uniquename>
    <vendor></vendor>
    .....
</compid>
<compid>Agent uninstall script
    <compversion>2.0
    <uniquename>Agent uninstall script</uniquename>
    <vendor>Sun Microsystems, Inc.</vendor>
    .....
</compid>
<compid>Agent installer resource bundle
    <compversion>2.0
    <uniquename>Agent installer resource
bundle</uniquename>
    <vendor>Sun Microsystems, Inc.</vendor>
    .....
</compid>
<compid>Agent Common Core and SDK
    <compversion>2.0
    <uniquename>Agent Common Core and SDK</uniquename>
    <vendor></vendor>
    .....
</compid>
<compid>SUNWames6
    <compversion>2.0
<uniquename>SUNWames6</uniquename>
    <vendor></vendor>
    .....
</compid>
<compid>Agent for ...
    <compversion>2.0
    <uniquename>Agent for ...</uniquename>
    <vendor></vendor>
    .....
</compid>
<compid>Sun ONE Identity Server Policy Agent
    <compversion>2.0
    <uniquename>Sun ONE Identity Server Policy
Agent</uniquename>
</compid>
```

Unable to uninstall Agent from Windows Start menu > Settings > Control Panel > Add/Remove Programs.

Possible Cause: Java's classpath may not be set correctly on the machine.

Solution: Use the following steps to uninstall the Agent.

1. Open Command Prompt Window.
2. Go to *Agent_Install_Dir*
3. Execute command:

```
java uninstall_Sun_ONE_Identity_Server_Policy_Agent
```

Lotus Domino 5.0.10

Domino Web Server starts with an error message "Unable to load filter".

Ensure that you have set the Domino DSAPI filter correctly. For steps on configuring the Domino DSAPI filter, see the section Configuring the Domino DSAPI Filter.

Domino DSAPI Filter is not functioning properly on the partitioned server.

Possible Cause: The database you have selected while configuring the DSAPI Filter may be wrong.

Solution: Ensure that you have selected the correct database while configuring the DSAPI filter.

Possible Cause: The partitioned database might not have been updated.

Solution: You might have to replicate the database from the Domino Admin Server.

Known Problems

IIS 5.0 Agent

After installing a Policy Agent on IIS 5.0, stopping individual web sites may occasionally lead to memory corruption messages. You can ignore these messages and restart the IIS server.

Agents are not effective after modification of Sun ONE Web Server configuration using the admin console.

The changes made by the agent installation to the server configuration files are overwritten by saving the changes in admin console.

The right procedure when using the admin console should be to load configuration first (from disk file to memory), then make modification, and save the changes (from memory to disk) by clicking the Apply button.

Policy Agents for Windows NT

Sun ONE Identity Server Policy Agents work in tandem with Sun ONE Identity Server to grant or deny user access to web servers in an enterprise. This chapter explains how URL access agents can be configured for IIS 4.0 running on the Windows NT operating system, version 4.0 with Service Pack 6.

Topics include:

- Before You Begin
- Installation Using the GUI
- Installation Using the Command-Line
- Using Secure Sockets Layer (SSL) with an Agent
- Setting the REMOTE_USER Server Variable
- Validating Client IP Addresses
- Shared Secret Encryption Utility
- Troubleshooting the IIS 4.0 Policy Agent
- Known Problems

Before You Begin

Be sure you're familiar with the concepts presented in Chapter 1, "Read This First." The chapter includes brief but important information on the following topics:

- How Policy Agents Work
- Java Runtime Environment 1.3.1

- The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers
- Configuring Agent for Multiple Web Server Instances on the Same Computer System
- Providing Failover Protection for Sun ONE Identity Server Agents
- Updating the Agent Cache
- Global Not-Enforced URL List
- Global Not-Enforced IP Address List
- Enforcing Authentication Only Without Enforcing Policies
- Forwarding LDAP User Attributes via HTTP Headers
- The AMAgent.properties File
- Setting the Fully Qualified Domain Name
- Configuring CDSSO

Supported Windows NT Web Server

The Sun ONE Policy Agent supports the following web server running on the Windows NT Server 4.0 SP6 operating system:

- Microsoft IIS 4.0

Installation Using the GUI

Installing the Policy Agent for Microsoft IIS 4.0

The IIS agent enforces policy on URL access for Microsoft's Internet Information Services (IIS 4.0) web server. The agent is an IIS ISAPI filter installed at the IIS web service level that will enforce policy on all IIS web sites. Technical considerations prevent the agent from being installed at the web site level.

Prior to installation, be sure that the entry for the system where the agent will be installed has a domain name set. If the web server that runs Sun ONE Identity Server is running on a separate system, make sure the server is also in the DNS query list.

To Install the Policy Agent on Microsoft IIS 4.0

You must have administrator privileges to run the installation program.

1. Unzip the product binaries.
2. Run the Installation program by double-clicking `setup.exe`.
3. In the Welcome window, click Next.
4. Read the License Agreement. Click Yes to accept the license agreement.
To search for the directory where you would like to install the agent, click Browse. To accept the default, click Next.
5. Enter the information about the web server where this agent will be installed:

Host Name: Enter the FQDN of the system where the Agent web server is installed. For example, `mycomputer.siroe.com`.

IIS Document Root: Enter the document root directory. This directory needs to be accessible by the web server root `w3svc`.

Server Port: Enter the port number for the web server that will be protected by the agent.

Server Protocol: If your web server has been configured for SSL, then select HTTPS; otherwise select HTTP.

Agent Deployment URI: Enter a directory name. The default Universal Resource Identifier (URI) is `/amagent`.

NOTE Agent uses value of `com.sun.am.policy.agents.agenturiprefix` property to support some essential functions such as notification and post data preservation. It is important to set a valid URL for this property. Its default value is:

`http://host.domain:port/agent_deployment_uri`

where *host*, *domain* and *port* are the FQDN and port number of the server where agent is installed. *agent_deployment_uri* is the prefix to URI which tells the web server where to look for agent's related HTML pages. Its default value is `amagent`.

When all the information is entered correctly, click Next.

6. Provide the following information about web server that runs Sun ONE Identity Server:

Primary Server Host: Enter the FQDN of the system where the primary web server that runs Sun ONE Identity Server is installed. For example, `myserver.siroe.com`.

Primary Server Port: Enter the port number for the web server that runs Sun ONE Identity Server.

Primary Server Protocol: If the web server that runs Sun ONE Identity Server is SSL-enabled, select HTTPS; otherwise select HTTP.

Primary Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Primary Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is `/amconsole`.

Failover Server Host: Enter the fully qualified name for the secondary web server that will run Sun ONE Identity Server if the primary web server becomes unavailable. If no failover host exists, then leave this field blank.

Failover Server Port: Enter the port number of the secondary web server that runs Sun ONE Identity Server. If no failover host exists, then leave this field blank.

Failover Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Agent Identity Server Shared Secret: Enter the password for the Identity Server internal LDAP authentication user password.

Re-enter Shared secret: Re-enter the password for the Identity Server internal LDAP authentication user password.

CDSSO Enabled: Check this box if you want to enable CDSSO feature.

CDSSO Component URL: Enter the CDSSO Component URL.

7. If all the information entered is correct, click Next.
8. Review your selections in the Summary panel and click Install Now.
9. When the installation is complete you may review the details, and then click Exit.
10. The installation modifies the system path by appending to it the location of the Agent libraries. In order for the change to take effect and for the Agent to work properly, you must reboot your computer.

NOTE If the IIS 4.0 agent was previously installed and uninstalled on your machine, you do not need to reboot if you are installing the same IIS 4.0 agent in the same directory.

Uninstalling and Disabling Policy Agents

When you no longer require the policy agent, you can uninstall it or disable it.

Uninstalling a Policy Agent

1. From the Start Menu, choose Settings > Control Panel.
2. In the Control Panel, open Add /Remove Programs.
3. In the Add/Remove Programs window, choose Sun ONE Identity Server Policy Agent.
4. Click Change/Remove.
5. Click Next on the Welcome panel.
6. Click Uninstall Now.
7. Click Close after uninstallation is complete.

Disabling a Policy Agent Installed on Microsoft IIS 4.0

1. Launch Internet Services Manager.
 - a. From the Start Menu, choose Programs > Windows NT 4.0 Options Pack > Microsoft Internet Information Server > Internet Services Manager.
2. Check the status of the filter.
 - a. In the left pane, right click the hostname of the computer and select Properties.
 - b. In the Master Properties section, select WWW Service, and click Edit.
 - c. Select the ISAPI Filters tab in the WWW Service Master Properties dialog that appears.
 - d. Highlight the filter named “Sun ONE Identity Server Policy Agent.”

Click on Edit to view the filter name and executable path. You will need this information when you want to re-enable the agent. Click Cancel to return to the program.
 - e. Click Remove
 - f. Click Apply and exit from the WWW Service Master Properties dialog.
3. Restart Microsoft IIS 4.0 by stopping `iisadmin`, and then starting `iisadmin` and `w3csvc`.

Installation Using the Command-Line

The command-line version of the Installation program provides you an alternative to the GUI version.

To Install an Agent Using the Command Line

1. In the directory where you unzipped the binaries, at the command line, enter the following command:

```
setup.bat -nodisplay
```
2. When prompted, provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement?

Install Sun One Policy Server Agent in this directory: Specify the directory where you want the agent to be installed. To accept the default directory that is displayed in brackets, press Enter. Otherwise, enter the full path.

3. When prompted, provide the following information about the web server instance this Agent will protect:

- Host Name
- Web Server Port
- Web Server Protocol
- Web Server Document Root
- Agent Deployment URI

For information about these items, refer to Part , “Installation Using the GUI” .”

4. When prompted, provide the following information about the web server that runs Sun ONE Identity Server Services:

- Primary Server Host
- Primary Server Port
- Primary Server Protocol
- Primary Server Deployment URI
- Primary Console Deployment URI
- Failover Server Host
- Failover Server Port
- Failover Server Deployment URI
- Failover Console Deployment URI
- Agent-Identity Server Shared secret
- Re-enter Shared secret
- CDSSO feature enabled
- CDSSO Component URL

For information about these items, refer to “Installation Using the GUI.”

5. When displayed, review the summary of installation information you’ve specified. Press Enter to continue, or enter exclamation mark (!) to exit the program.
6. The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation

What would you like to do
```

When prompted, **What would you like to do?**, enter 1 to start the installation.

7. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Installed	Available
2. Done		

To see log information, enter 1. To exit the Installation program, enter 2.

8. When installation is complete, restart IIS 4.0.

To Uninstall an Agent Using the Command Line

1. In the *Agent_Install_Dir* directory, at the command line, enter the following command:

```
java uninstall_Sun_ONE_Identity_Server_Policy_Agent -nodisplay
```

2. When prompted, provide the following information:

The uninstaller detects agents that were previously installed using the `setup` program. Enter 1 to uninstall the agent.

3. The following message is displayed:

```
1. Uninstall Now
2. Start Over
3. Exit Uninstallation
What would you like to do?
```

When prompted, **What would you like to do?**, enter 1 to start the uninstallation.

4. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Full	Available
2. Done		

To see log information, enter 1. To exit the uninstallation program, enter 2.

5. When uninstallation is completed, reboot the system.

If you want to see more details of the uninstallation, a log file is written in the following location:

```
%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall.*
```

Using Secure Sockets Layer (SSL) with an Agent

During installation, if you choose the HTTPS protocol, the agent is automatically configured and ready to communicate over SSL.

NOTE Before proceeding with the following steps, ensure that the Web Server is configured for SSL.

The Agent's Default Trust Behavior

By default, the policy agent installed on a remote Microsoft IIS 4.0 will trust any server certificate presented over SSL by the web server that runs Sun ONE Identity Server services; the agent does not check the root Certificate Authority (CA) certificate. If the web server that runs Sun ONE Identity Server is SSL-enabled, and you want the policy agent to perform certificate-checking, you must do two things:

1. Disable the agent's default trust behavior.
2. Install a root CA certificate on the remote web server where the agent is installed. The root CA certificate must be the same one that is installed on the web server that runs Sun ONE Identity Server.

Disabling the Agent's Default Trust Behavior

The following property exists in the `AMAgent.properties` file, and by default it is set to `true`:

```
com.sun.am.policy.agents.trustServerCerts=true
```

This means that the agent does not perform certificate-checking.

To Disable the Default Behavior

The following property must be set to `false`:

```
com.sun.am.policy.agents.trustServerCerts=false
```

Installing the Root CA Certificate on the Remote Web Server

The root CA certificate that you install on the remote web server must be the same one that is installed on the web server that runs Sun ONE Identity Server.

To Install the Root CA Certificate on Microsoft IIS

1. Go to the following directory:

Agent_Install_Dir\iis\cert

2. Add the same root certificate that is installed on the web server that runs Sun ONE Identity Server into the existing certificate database. At the command line, enter the following command:

```
\Agent_Install_Dir\bin\certutil -A -n cert-name -t "C,C,C" -d cert-dir -i cert-file
```

using the following variables:

- *cert-name* can be any name for this root certificate.
- *cert-dir* is directory where the certificate-related files are located. On Windows NT the location is

```
Agent_Install_Dir\bin
```

- *cert-file* is the base-64 encoded root certificate file.
- For more information on certutil, type `certutil -H`

To verify that the root certificate was installed properly in the certificate database, enter the following command:

```
Agent_Install_Dir\bin\certutil -L -d .
```

You should see the root certificate added and listed in the output of the command.

3. Restart IIS.

Setting the REMOTE_USER Server Variable

The REMOTE_USER server environment variable can be set to a Sun ONE Identity Server authenticated user or anonymous user. By setting this variable to a specific user, the user becomes available to web applications (such as a CGI, servlet, or ASP program). This feature makes it possible to personalize the content of displayed HTML pages to specific users.

To enable the REMOTE_USER feature, perform the following steps:

1. From the Windows Start menu, choose Programs > Windows NT Option Pack > Microsoft Internet Information Server > Internet Services Manager.

This will launch the Microsoft Management Console.

2. On the web site that you want the Sun ONE Identity Server agent to protect, select Properties.
3. Select the Directory Security tab.

4. In the Anonymous Access and Authentication Control section, click Edit, and select Allow Anonymous Access (selected by default), Basic Authentication (not selected by default), and deselect Challenge/Response (selected by default).

`REMOTE_USER` will be set while accessing allowed URLs.

To enable the `REMOTE_USER` setting for globally not-enforced URLs as specified in the `AMAgent.properties` file (these are URLs that can be accessed by unauthenticated users), you must set the following property in the `AMAgent.properties` file to `TRUE` (by default, this value is set to `FALSE`):

```
com.sun.am.policy.agents.anonRemoteUserEnabled=TRUE
```

When you set this property value to `TRUE`, the value of `REMOTE_USER` will be set to the value contained in the following property in the `AMAgent.properties` file (by default, this value is set to `anonymous`):

```
com.sun.am.policy.agents.unauthenticatedUser=anonymous
```

Validating Client IP Addresses

This feature can be used to enhance security by preventing the stealing or *hijacking* of SSO tokens.

The `AMAgent.properties` file contains a property titled `com.sun.am.policy.agents.client_ip_validation_enable`, which by default, is set to `false`.

If you set this property value to `true`, client IP address validation will be enabled for each in-coming request that contains an SSO token. If the IP address from which request was generated does not match the IP address issued for the SSO token, the request will be denied. This is essentially the same as enforcing a deny policy.

This feature should not be used, however, if the client browser uses a web proxy or if there is a load-balancing application somewhere between the client browser and the agent-protected web server. In such cases, the IP address appearing in the request will not reflect the real IP address on which the client browser runs.

Shared Secret Encryption Utility

The Sun ONE Identity Server Policy Agent stores the shared secret in the `AMAgent.properties` file. By default, this password is the Identity Server internal LDAP authentication user password. This can be changed on the server side by editing the file `AMConfig.Properties`.

The property `com.sun.am.policy.am.password` in the file `AMConfig.Properties` is set with the encrypted shared secret while installing the agent.

To reset or change the shared secret, you can use the following utility and set the value in the property.

1. Go to the following directory:

`Agent_Install_Dir\bin`

2. Execute the following script from the command line:

`cryptit shared_secret`

3. Cut and paste the output from Step 2 in the property:

`com.sun.am.policy.am.password`

4. Restart the Web Server and try accessing any resource protected by the agent. If the agent gets redirected to the Identity Server, this indicates the above steps were executed properly.

Troubleshooting the IIS 4.0 Policy Agent

If you are experiencing problems with your installation try the following:

- Check the installation log file for errors:
`%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall.nnnn`
- Re-install by uninstalling and then installing.
- Verify agent loading in IIS:
 - a. Launch Internet Services Manager.
 - b. From the Start menu, choose Programs > Windows NT Option Pack > Microsoft Internet Information Server > Internet Services Manager.

This will launch the Microsoft Management Console.

- c. In the left window, right click on the hostname of the computer and select Properties.
- d. In the Master Properties section, select WWW Service and click Edit.
- e. Select the ISAPI Filters tab in the WWW Service Master Properties dialog that appears.
- f. Look for the filter name “Sun ONE Identity Server Agent.”

If the Filter name “Sun ONE Identity Server Agent” does not appear at all, then check that the installer was run, and look for any errors during installation. The install log is located at:

```
%TEMP%\Sun_ONE_Identity_Server_Policy_Agent_uninstall.nnnn
```

A green arrow pointing up in the Status column to the right of the “Sun ONE Identity Server Agent” indicates the agent loaded successfully into IIS. A red arrow pointing down indicates that the filter failed to load. The most likely cause of the filter not loading successfully (red arrow) is that it cannot locate the required dll files.

- g. Check your system path to ensure that the following directory is present:

```
# \Agent_Install_Dir\bin
```

- h. If the filter did not load successfully check the following:

- Check the path of the Agent DLL by clicking “Sun ONE Identity Server Agent” and then Edit. Ensure that the path in the text box labeled Executable is valid.
- The agent also needs several DLL files. Check that the following files exist in the lib directory:

```
amsdk.dll
ames6.dll
libnspr4.dll
libplc4.dll
libplds4.dll
libxml2.dll
nss3.dll
ssl3.dll
```

If the libraries are in your system path, try rebooting the system.

- IIS logs filter loading errors in the System Event Log. To check the event log:
 - a. From the Start menu, choose Programs > Administrative Tools > Event Viewer.
 - b. Select the System Log.
 - c. Check for Error messages with Source W3SVC.
- If the agent loads, but returns HTTP 500 Internal Server Error for all URL requests to the IIS web server, it indicates that the agent has loaded, but did not properly initialize. Returning HTTP 500 Internal Server Error for all HTTP requests is a fail-safe to protect URL resources when the Agent cannot initialize. The most likely cause is a Sun ONE Identity Server agent or server misconfiguration or unavailability.
- Check the agent debug log.

The log is located by default at the *Agent_Install_Dir* directory. This is the best source of debug information for resolving initialization and agent operation issues. The log file directory is specified by the property:

`com.sun.am.policy.am.logFile`

In the `AMAgent.properties` file located in the directory:

Agent_Install_Dir\iis\config_PathInstanceName

Agent_Install_Dir\iis\config_PathInstanceName

The property `com.sun.am.policy.am.loglevels` controls the verbosity of the log information. Set the logging level for the specified logging categories.

The format of the values is:

ModuleName [: *Level*] , *ModuleName* [: *Level*]] *

The currently used module names are AuthService, NamingService, PolicyService, SessionService, PolicyEngine, ServiceEngine, Notification, PolicyAgent, RemoteLog, and all. If the level is omitted, then the logging module will be created with the default logging level, which is the logging level associated with the 'all' module.

The all module can be used to set the logging level for all modules. This will also establish the default level for all subsequently created modules. The meaning of the 'Level' value is described below:

The property `com.sun.am.policy.am.loglevels` controls the verbosity of the log information. The meaning of the 'Level' value is described below:

0 = Disable logging from specified module*

1 = Log error messages

2 = Log warning and error messages

3 = Log info, warning, and error messages

4 = Log debug, info, warning, and error messages

5 = Like level 4, but with even more debugging messages.

- Check that the agent can locate the `AMAgent.properties` configuration file.

The agent uses the registry key

`HKEY_LOCAL_MACHINE\Software\Sun Microsystems\Identity Server IIS Agent` to locate the `AMAgent.properties` file. The `AMAgent.properties` file is located at:

`Agent_Install_Dir\iis\config_PathInstanceName`

- The agent uses the Application Event Log to log errors that occur before the debug log file specified in `AMAgent.properties` is started.
 - a. From the Start menu, choose Programs > Administrative Tools > Event Viewer.
 - b. Select the Application Log.
 - c. Check for Error messages with Source Sun ONE Identity Server IIS Agent.

Cannot install Agent after previous installation is removed

The following is an example message displayed when you run the Agent installer:

```
"Sun ONE Identity Server Policy Agent 2.0 for Microsoft Internet
Information Services is installed. Please refer to installation
manual to configure this agent for another web server instance.
Or uninstall it before installing another agent."
```

Possible Causes:

- You might have an existing installation of Agent.
- You might have a previously installed Agent and did not use Agent's uninstaller to uninstall the Agent.
- The installer's `productregistry` file may be corrupted.

Solution:

- Check that you have uninstalled any existing installation of Agent.
- The `productregistry` file may be corrupted if there is no existing installation of Agent. This file is used by the installer to track installed products. It is found in `C:\WINT\system32` directory.

NOTE Make a backup copy of this file before you make changes.

Remove the Agent product entry in this file. This entry starts with the following lines:

```
<compid>SUNWamcom
  <compversion>2.0
  <uniqueid>SUNWamcom</uniqueid>
  <vendor></vendor>
  .....
</compid>
<compid>Agent uninstall script
  <compversion>2.0
  <uniqueid>Agent uninstall script</uniqueid>
  <vendor>Sun Microsystems, Inc.</vendor>
  .....
</compid>
<compid>Agent installer resource bundle
  <compversion>2.0
  <uniqueid>Agent installer resource
bundle</uniqueid>
  <vendor>Sun Microsystems, Inc.</vendor>
  .....
</compid>
<compid>Agent Common Core and SDK
  <compversion>2.0
  <uniqueid>Agent Common Core and SDK</uniqueid>
  <vendor></vendor>
  .....
</compid>
<compid>SUNWames6
  <compversion>2.0
<uniqueid>SUNWames6</uniqueid>
  <vendor></vendor>
  .....
</compid>
<compid>Agent for ...
  <compversion>2.0
  <uniqueid>Agent for ...</uniqueid>
  <vendor></vendor>
  .....
</compid>
<compid>Sun ONE Identity Server Policy Agent
  <compversion>2.0
```

```

        <uniquename>Sun ONE Identity Server Policy
Agent</uniquename>
    </compid>

```

Unable to uninstall Agent from Windows Start menu > Settings > Control Panel-> Add/Remove Programs.

Possible Cause: Java's classpath may not be set correctly on the machine.

Solution: Use the following steps to uninstall the Agent:

1. Open command prompt window
2. Go to the directory where Agent is installed
3. Execute `java uninstall_Sun_ONE_Identity_Server_Policy_Agent`

Known Problems

Internet services hang when you attempt to shutdown individual websites.

Workaround

It is highly recommended that you do not shut down your websites individually. Instead, you should shut them down collectively by stopping the IIS Admin Service, restarting the IIS Admin Service, and then restarting the individually managed services.

1. To stop the IIS Admin Service, issue the following command from the command line:

```
c:\>net stop iisadmin /y
```

Alternatively, you can shutdown the IIS Admin Service from the Services menu:

- a. From the Start menu, choose Settings > Control Panel.
- b. Click on Services.
- c. Select IIS Admin Service.
- d. Click Stop.

This will shutdown all internet services managed by the `iisadmin` process, including FTP services, WWW services and SMTP services.

2. To restart the ISS Admin Service, issue the following command from the command line:

```
c:\>net start iisadmin
```

Alternatively, you can restart the services from the Services menu:

- a. From the Start menu, choose Settings > Control Panel.
 - b. Click on Services.
 - c. Select IIS Admin Service.
 - d. Click Start.
3. To restart the individual services, issue the following command from the command line:

```
c:\>net start w3svc
```

Alternatively, you can restart the individual services from the Services menu:

- a. From the Start menu, choose Settings > Control Panel.
- b. Click on Services.
- c. Select World Wide Web Publishing.
- d. Click Start.

IIS 4.0 stop problem

After installing a Policy Agent on IIS 4.0, stopping individual web sites may occasionally lead to memory corruption messages. You can ignore these messages and restart the IIS server

Policy Agent for Red Hat Linux 7.2

Sun ONE Identity Server Policy Agents work in tandem with Sun ONE Identity Server to grant or deny user access to web servers in an enterprise. This chapter explains how to install the Sun ONE Identity Server Policy Agent for Apache 1.3.26 server running on the Red Hat Linux 7.2 operating system.

Topics include:

- Before You Begin
- Configuring Apache Web Server with Posix Threads
- Installation Using the GUI
- Installation Using the Command-Line
- Configuring Agent for Multiple Web Server Instances
- Using Secure Sockets Layer (SSL) with an Agent
- Setting the REMOTE_USER Server Variable
- Validating Client IP Addresses
- Shared Secret Encryption Utility
- Troubleshooting Information

Before You Begin

Be sure you're familiar with the concepts presented in Chapter 1, "Read This First." The chapter includes brief but important information on the following topics:

- How Policy Agents Work
- Java Runtime Environment 1.3.1

- The Web Server that Runs Sun ONE Identity Server Services vs. Remote Web Servers
- Configuring Agent for Multiple Web Server Instances on the Same Computer System
- Providing Failover Protection for Sun ONE Identity Server Agents
- Updating the Agent Cache
- Global Not-Enforced URL List
- Global Not-Enforced IP Address List
- Enforcing Authentication Only Without Enforcing Policies
- Forwarding LDAP User Attributes via HTTP Headers
- The AMAgent.properties File
- Setting the Fully Qualified Domain Name
- Configuring CDSSO

Configuring Apache Web Server with Posix Threads

Before installing the agent, you must complete the following tasks in the order they are listed below to make sure Apache Web Server is configured with Posix Threads library. Failing to perform these tasks may result in the application becoming unusable or may result in the entire system becoming unstable and unusable.

1. Get the Apache source from <http://httpd.apache.org/>
2. Edit the `Configure` file located in the directory:
`/Apache_root/src`
3. Search for `linux22` in the `Configure` file.
4. Append `-lpthread` after `-lm` to the `LIBS` variable.

```
*-linux22)
    # This handles linux 2.2 and above (2.4, ...)
    DEF_WANTHSREGEX=yes
    OS='Linux'
```

```
CFLAGS="$CFLAGS -DLINUX=22"
LIBS="$LIBS -lm -lpthread"
```

5. Save and close the file.
6. Run the `configure` script located in the directory:
`/Apache_root/`
7. Rebuild and install Apache Web Server.
8. Install the Apache Agent.

Installation Using the GUI

Installing the Policy Agent

You must have root permissions when you run the agent installation program.

1. Unpack the product binaries using the following command:

```
# gunzip -dc agent_Linux_apache.tar.gz | tar -xvof -
```
2. Run the `setup` program. You'll find the program in the directory where you untarred the binaries. At the command line, enter the following:

```
# ./setup
```
3. In the Welcome page, click Next.

4. Read the License Agreement. Click Yes to agree to the license terms.
To search for the directory where you would like to install the agent, click Browse. To accept the default, click Next.
5. When prompted, provide the following information about the web server this agent will protect:

Install Sun ONE Identity Server Policy Agent in this directory: Enter the full path to the directory where you want this agent to be installed, and then click Next.

Host Name: Enter the FQDN of the machine where the web server is installed. For example, `mycomputer.siroe.com`

Apache Configuration Directory: Specify the Apache server configuration directory where the `httpd.conf` file is located.

Web Server Port: Enter the port number for the web server that will be protected by the agent.

Web Server Protocol: If the web server has been configured for SSL, choose HTTPS; otherwise choose HTTP.

Agent Deployment URI: Enter a directory name. The default Universal Resource Identifier (URI) is `/amagent`.

SSL Ready: You should select this option if the Apache web server you are using has support for SSL. Your Apache web server is considered SSL ready if it has support for `mod_ssl` and its sources have been compiled using EAPI rule.

To find out if your Apache web server has been compiled with the EAPI flag, go to the `bin` directory of the Apache web server and type the command:

```
# ./httpd -V
```

You can see various flags that the Apache web server was compiled with. If the flag `-D EAPI` is displayed in this list, it indicates that your Apache Web Server is SSL ready. However, if you do not see this flag; it does not necessarily indicate that the Web Server does not have support for `mod_ssl`.

The supported configuration for Apache web server are:

- a. Apache Web Server without `mod_ssl` support
- b. Apache Web Server with `mod_ssl` and EAPI flag enabled.

NOTE Apache Web Server with `mod_ssl` support and EAPI flag disabled configuration is not supported by Sun ONE Identity Server Policy Agents.

When all the information is entered correctly, click Next.

6. Enter information about the web server that runs Sun ONE Identity Server policy and management. The Policy Agent will connect to this server.

Primary Server Host: Enter the FQDN of the system where the primary web server that runs Sun ONE Identity Server is installed. For example, `myserver.siroe.com`.

Primary Server Port: Enter the port number for the web server that runs Sun ONE Identity Server.

Primary Server Protocol: If the web server that runs Sun ONE Identity Server is SSL-enabled, select HTTPS; otherwise select HTTP.

Primary Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Primary Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is `/amconsole`.

Failover Server Host: Enter the FQDN for the secondary web server that will run Sun ONE Identity Server if the primary web server becomes unavailable. If no failover host exists, then leave this field blank.

Failover Server Port: Enter the port number of the secondary web server that runs Sun ONE Identity Server. If no failover host exists, then leave this field blank.

Failover Server Deployment URI: Enter the location that was specified when Sun ONE Identity Server was installed. The default URI for Sun ONE Identity Server is `/amserver`.

Failover Console Deployment URI: Enter the location that was specified when Sun ONE Identity Server console was installed. The default URI for Sun ONE Identity Server is `/amconsole`.

Agent Identity Server Shared Secret: Enter the password for the Identity Server internal LDAP authentication user.

Re-enter Shared secret: Re-enter the password for the Identity Server internal LDAP authentication user.

CDSSO Enabled: Check this box if you want to enable CDSSO feature.

CDSSO Component URL: Enter the CDSSO Component URL.

When all the information is entered correctly, click Next.

7. Review the Installation Summary to be sure that the information you've entered is correct. If you want to make changes, click Back. If all the information is correct, click Next
8. In the Ready to Install page, click Install Now.
9. When the installation is complete, you can click Details to view details about the installation, or click Close to end the Installation program.
10. You must restart your Apache web server for the installation to be complete.

Uninstalling the Policy Agent

Use the following steps to uninstall the policy agent:

1. In the directory where the agent is installed, at the command line, enter the following command:

```
# ./uninstall_linux_apache_agent
```
2. Click Next on Welcome panel.
3. Click Uninstall Now.
4. Click Close after uninstallation is complete.

Installation Using the Command-Line

The command-line version of the Installation program provides you an alternative to the GUI version.

Installing the Policy Agent

You must have root permissions when you run the agent installation program.

1. Unzip the Solaris tar file using the following command:

```
# gunzip -dc agent_Linux_apache.tar.gz | tar -xvof -
```

2. Run the `setup` program. You'll find the program in the directory where you untarred the binaries. At the command line, enter the following:

```
# ./setup -nodisplay
```

3. When prompted, provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement? Enter yes.

Install Sun ONE Identity Server Agent in this directory: Enter the full path to the directory in which you want to install the policy agent.

4. Provide the following information about the Web Server this agent will protect:

- o Web Server Host Name
- o Apache Configuration Directory
- o Web Server Port
- o Web Server Protocol
- o Agent Deployment URI
- o SSL Ready

For more information on each of these items, see “Installing the Policy Agent.”

5. Provide the following information about the web server that runs Sun ONE Identity Server:

- o Primary Server Host
- o Primary Server Port
- o Primary Server Protocol
- o Primary Server Deployment URI
- o Primary Console Deployment URI
- o Failover Server Host
- o Failover Server Port
- o Failover Server Protocol
- o Failover Server Deployment URI
- o Failover Console Deployment URI

- Agent-Identity Server Shared secret
- Re-enter Shared secret
- CDSSO Enabled
- CDSSO Component URL

For more information on each of these items, see “Installing the Policy Agent.”

6. The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation
```

When prompted, **What would you like to do?**, enter 1 to start the installation.

7. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Agent	Installed	Available
2. Done		

To see log information, enter 1. To exit the Installation program, enter 2.

Uninstalling the Policy Agent

1. In the *Agent_Install_Dir* directory, at the command line, enter the following command at the command line:

```
# ./uninstall_linux_apache_agent -nodisplay
```

2. The following text is displayed:


```
The uninstaller has detected the following agents on this system:
1. Agent 2.0 for Apache 1.3.26 [/usr/local]
2. Exit
Please select an installed agent from the following list:
```

Enter 1, to remove the product.

3. The following text is displayed:

```
Ready to Uninstall

1. Uninstall Now
2. Start Over
3. Exit Uninstallation
```

When prompted, **What next?** enter 1 to begin uninstallation.

4. The following text is displayed:

Product	Result	More Information
1. Sun ONE Identity Server Policy Agent	Full	Available
2. Done		

To see log information on the agent, enter 1. To exit the uninstallation program, enter 2.

Configuring Agent for Multiple Web Server Instances

To configure an Agent for multiple web server instances on a single computer, use the GUI or the command-line version of the Agent Installation program to install the first agent. After the first agent is installed, you can then install successive agents using the `config` script. This script must be run from the command line as described in the next section.

To Configure Agent for Multiple Web Server Instances on the Same Computer System

Once you have installed an agent on your system, you can install successive agents on that system using a script that is copied to the system during the agent installation. The two scripts, `config_linux` and `unconfig_linux`, located in the following directory:

Agent_Install_Dir/agents/apache/bin

To install additional agents on a system after the original agent has been installed, run the `config_linux` script from the `bin` directory using the following command:

```
# ./config_linux
```

Follow the prompts to install the additional agents. For information on each of the prompts, see “Installation Using the GUI.” In general, information needs to be entered for both the protected Apache server instance and the Sun ONE Identity Server server(s). The following text shows an example run:

```
# ./config_linux
Enter the Apache Server Configuration Directory:
[/etc/httpd/conf]
SSL Ready: [true] false
Enter the Local Hostname: [mycomputer.siroe.com]
Enter the Agent Web Server Port: [80]
Select Agent Web Server Protocol: [1] http [2] https-->[1]
Enter the Agent Deployment URI: [/amagent]
Select Identity Server Protocol: [1] http [2] https --> [1]
Enter the Identity Server Hostname: [mycomputer.siroe.com]
Enter the Identity Server Port: [58080]
Enter the Identity Server Deployment URI [/amserver]
Enter the Identity Server's Console Deployment URI [/amconsole]
Enter the Identity Server's Console Deployment URI [/amconsole]
Select Failover Identity Server Protocol: [1] http [2] https [3]
no failover --> []
```

```

Enter the Failover Identity Server Hostname:
[]mycomputer.siroe.com
Enter the Failover Identity Server Port: []
Enter the Identity Server Deployment URI [/amserver]
Enter the Identity Server's Console Deployment URI [/amconsole]
Enter Agent-Identity Server shared secret:
Re-enter Agent-Identity Server shared secret:
Configuring Apache Web Server ...
Done

```

NOTE The `config` script does not allow you to enter CDSSO details. You must configure it manually. For more information, see “Configuring CDSSO.”

Using the `config` Script for Silent Installations

The `config_linux` script can also be used to do silent and non-interactive agent installations. For details on its usage, use the `config_linux -h` command to display details:

```

# ./config_linux -h
Usage: config [ -r response_file | -R | -h ]
    -r specifies a response file.
    -R prints out the response file template.
    -h prints out this message.

```

In order to perform a silent installation, you must supply a *response.file* for each of the agents you want to install. The command `config_linux -R` indicates the fields which you must supply in the *response.file*. This text file needs to be prepared before you begin the silent installation.

```

./config_linux -R
Response file contains:
AGENT_PROTOCOL          # agent protocol: http|https
AGENT_HOST              # agent hostname
AGENT_PORT              # agent server port
AGENT_DEPLOY_URI        # agent deploy URI

```

```

FAILOVER_SERVER_HOST      # failover identity server name
FAILOVER_SERVER_PORT      # failover identity server port
FAILOVER_SERVER_DEPLOY_URI # failover identity server deploy URI
FAILOVER_CONSOLE_DEPLOY_URI # failover identity server console deploy URI
PRIMARY_SERVER_HOST       # primary identity server name
PRIMARY_SERVER_PORT       # primary identity server port
PRIMARY_SERVER_PROTO      # primary identity server protocol: http|https
PRIMARY_SERVER_DEPLOY_URI # primary identity server deploy URI
PRIMARY_CONSOLE_DEPLOY_URI # primary identity server console deploy URI
SHARED_SECRET             # shared secret between agent and DSAME server
SERVER_INSTANCE           # web server instance directory
NOTIFICATION_ENABLE       # notification enabled
AGENT_URL_CASE_IGNORE     # url comparison case ignore

```

Here is an example for *response.file*, named `response.apache`.

```

AGENT_PROTOCOL=http
AGENT_HOST=mycomputer.siroe.com
AGENT_PORT=80
AGENT_DEPLOY_URI=/amagent
FAILOVER_SERVER_HOST=failover_computer.siroe.com
FAILOVER_SERVER_PORT=58080
FAILOVER_SERVER_DEPLOY_URI=/amserver
FAILOVER_CONSOLE_DEPLOY_URI=/amconsole
PRIMARY_SERVER_HOST=primary_computer.siroe.com
PRIMARY_SERVER_PORT=58080
PRIMARY_SERVER_PROTO=http
PRIMARY_SERVER_DEPLOY_URI=/amserver
PRIMARY_CONSOLE_DEPLOY_URI=/amconsole
SHARED_SECRET=encrypted_shared_secret
SERVER_INSTANCE=/Agent_Install_Dir/apache26/conf
NOTIFICATION_ENABLE=true
AGENT_URL_CASE_IGNORE=true

```

Below is an example showing how the `config_linux` script is used in conjunction with the `response.apache` file to complete a silent installation.

```

# ./config_linux -r response.apache
Configuring Apache Web Server ...
done

```

NOTE Be sure to use the `unconfig_linux` script to uninstall any agent that was installed using the `config_linux` script—you cannot use the GUI installation program to uninstall agents that were installed from the command line. The GUI uninstaller must be executed only after unconfiguring all the existing agents installed using command-line `unconfig` script.

Removing an Agent Using the `unconfig` Script

To remove an agent that was installed from the command line using the `config_linux` script, use the script `unconfig_linux`. The `unconfig_linux` script is located in the following directory:

Agent_Install_Dir/agents/apache/bin

Here is an example run of the `unconfig_linux` script.

```
# ./unconfig_linux /web_server_root/httpd/conf
Unconfiguring webserver ... \c
done.
```

Using Secure Sockets Layer (SSL) with an Agent

During installation, if you choose the HTTPS protocol, the agent is automatically configured and ready to communicate over SSL.

NOTE Before proceeding with the following steps, ensure that the Web Server is configured for SSL.

The Agent's Default Trust Behavior

By default, policy agent installed on a remote Apache Server 1.3.26 will trust any server certificate presented over SSL by the Web Server that runs Sun ONE Identity Server; the agent does not check the root Certificate Authority (CA) certificate. If the Web Server that runs Sun ONE Identity Server is SSL-enabled, and you want the policy agent to perform certificate-checking, you must do the following:

1. Disable the agent's default trust behavior.

2. Install a root CA certificate on the remote web server (where the agent is installed). The root CA certificate must be the same one that is installed on the web server that runs Sun ONE Identity Server service.

Disabling the Agent's Default Trust Behavior

The following property exists in the `AMAgent.properties` file, and by default it is set to `true`:

```
com.sun.am.policy.amcpa.trustServerCerts=true
```

This means that the agent does not perform certificate-checking.

To Disable the Default Behavior

The following property must be set to `false`:

```
com.sun.am.policy.amcpa.trustServerCerts=false
```

Installing the Root CA Certificate on the Remote Web Server

The root CA certificate that you install on the remote web server must be the same one that is installed on the web server that runs Sun ONE Identity Server.

To Install the Root CA Certificate on Apache 1.3.26

You can use the `certutil` program to install the root CA Certificate on Apache 1.3.26.

1. In C shell, at the command line, enter the following commands (assuming `/etc/httpd/apache` is the directory where the Apache config file is located):

```
# cd /etc/apache/cert
# setenv LD_LIBRARY_PATH
  /Agent_Install_Dir/agents/apache/lib:/Agent_Install_Dir/agents/lib
```
2. Create the necessary certificate database if you have not already done.

```
# /Agent_Install_Dir/agents/apache/cert/certutil -N -d .
```
3. Install root CA certificate.

```
# /Agent_Install_Dir/agents/apache/cert/certutil -A -n cert-name -t
"C,C,C" -d cert-dir -i cert-file
```

In the commands above, the variables represent the following:

- *cert-name* can be any name for this root certificate.
- *cert-dir* is directory where the certificate-related files are located.
- *cert-file* is the base-64 encoded root certificate file.

For more information on the `certutil` utility, enter `certutil -H` for online Help.

4. To verify that the certificate is properly installed, at the command line, enter the following:

```
# ./certutil -L -d .
```

Trust database information will display including the name of the root CA certificate you installed. Example:

Certificate Name	Trust Attributes
<i>cert-name</i>	C,C,C
p	Valid peer
P	Trusted peer (implies c)
c	Valid CA
T	Trusted CA to issue client certs (implies c)
C	Trusted CA to certs(only server certs for ssl) (implies c)
u	User cert
w	Send warning

Setting the REMOTE_USER Server Variable

The `REMOTE_USER` server environment variable can be set to a Sun ONE Identity Server authenticated user or anonymous user. By setting this variable to a specific user, the user becomes available to web applications (such as a CGI, servlet, or ASP program). This feature makes it possible to personalize the content of displayed HTML pages to specific users.

To enable the `REMOTE_USER` setting for globally not-enforced URLs as specified in the `AMAgent.properties` file (these are URLs that can be accessed by unauthenticated users), you must set the following property in the `AMAgent.properties` file to `TRUE` (by default, this value is set to `FALSE`):

```
com.sun.am.policy.agents.anonRemoteUserEnabled=TRUE
```

When you set this property value to `TRUE`, the value of `REMOTE_USER` will be set to the value contained in the following property in the `AMAgent.properties` file (by default, this value is set to `anonymous`):

```
com.sun.am.policy.agents.unauthenticatedUser=anonymous
```

Validating Client IP Addresses

This feature can be used to enhance security by preventing the stealing or *hijacking* of SSO tokens.

The `AMAgent.properties` file contains a property titled `com.sun.am.policy.agents.client_ip_validation_enable`, which by default is set to `false`.

If you set this property value to `true`, client IP address validation will be enabled for each in-coming request that contains an SSO token. If the IP address from which request was generated does not match the IP address issued for the SSO token, the request will be denied. This is essentially the same as enforcing a deny policy.

This feature should not be used, however, if the client browser uses a web proxy or if there is a load-balancing application somewhere between the client browser and the agent-protected web server. In such cases, the IP address appearing in the request will not reflect the real IP address on which the client browser runs.

Shared Secret Encryption Utility

The Policy Agent stores the shared secret in the `AMAgent.properties` file. By default, this password is the Identity Server internal LDAP authentication user password. This can be changed on the server side by editing the `AMConfig.Properties` file.

The property `com.sun.am.policy.am.password` in the `AMConfig.Properties` file is set with the encrypted shared secret while installing the agent.

To reset or change the shared secret, you can use the following utility and set the value in the property.

1. Go to the following directory:

```
Agent_Install_Dir/bin
```

2. Execute the following script from the command line:

```
crypt_util shared_secret
```

3. Cut and paste the output from Step 2 in the property:

```
com.sun.am.policy.am.password
```

4. Restart the Web Server and try accessing any resource protected by the agent. If the agent gets redirected to the Sun ONE Identity Server, this indicates the above steps were executed properly.

Troubleshooting Information

Error message displayed during startup

Apache server displays the following error message during startup after the agent is installed:

```
Syntax error on line 1 of
/etc/opt/SUNWam/agents/apache/config/_usr_local_apache_conf/dsamen.conf:

Invalid command 'LoadModule', perhaps mis-spelled or defined by a
module not included in the server configuration

./apachectl start: httpd could not be started
```

Solution: This indicates that the Apache server does not have `mod_so` enabled and consequently does not support dynamic shared objects. To enable `mod_so` support, refer to Apache server documentation at <http://httpd.apache.org/>

J2EE Agents

Chapter 6, “Read This First”

Chapter 7, “Policy Agent for WebLogic 6.1 SP2”

Chapter 8, “Policy Agent 2.0 for IBM WebSphere 4.0.4 AE”

Chapter 9, “Policy Agent for Sun ONE Application Server 7.0”

Appendix A, “Configuration Tasks Performed by Installer”

Appendix B, “Sample Scenarios for Role-to-Principal Mapping”

Appendix C, “Using the Policy Agent Debug Engine”

Read This First

The Sun ONE Identity Server Policy Agent enables Application Servers to enforce authentication and authorization using Sun ONE Identity Server services, thereby securing client access to the hosted J2EE applications and enforcing J2EE security policies defined in the deployed application's Deployment Descriptors.

This chapter provides a brief overview of Sun ONE Identity Server Policy Agent for Application Server, as well as some concepts you will need to understand before proceeding with the Installation program.

Topics include:

- Uses of Policy Agent for Application Server
- Supported Servers

Uses of Policy Agent for Application Server

The Sun ONE Identity Server Policy Agent for Application Server may be installed for protecting a variety of hosted J2EE applications which may require a varying set of security policy implementation. The security infrastructure of J2EE provides declarative as well as programmatic security which are platform independent and are supported by all the compliant J2EE application servers. For details on how to use J2EE platform's declarative as well as programmatic security, refer to J2EE documentation which can be found at <http://java.sun.com/j2ee>.

The Agent provides the ability to enable role-to-principal mapping for protected J2EE applications with Sun ONE Identity Server principals. Thus at runtime, when a J2EE policy is evaluated, it is done against the information available in Sun ONE Identity Server. Using this functionality, administrators may configure their hosted J2EE applications to be protected by the Agent which provides real security services and also other key features such as single sign-on.

Examples

A Commerce Application

A commerce application may have a variety of specialized Enterprise JavaBeans components that offer a spectrum of services to the clients. For instance, there could be a specialized component that provides the ability to create purchase orders. Similarly, there could be a specialized component that provides the ability to approve a purchase order. While such components provide the basic business services for the application to function, the very nature of tasks that they accomplish require a security policy to enforce appropriate use of such services.

Using the deployment descriptors, the application vendor or developer can express intent by protecting such components using abstract security role names. For example, there could be a role called “Buyer” which protects the component that provides the ability to create a purchase order. Similarly, there could be a role called “Approver” which protects the component that provides the ability to approve a purchase order. While these roles convey the intent of the application vendor or developer to enforce such security policies, they will not be useful unless these abstract role names are mapped to real life principals such as actual users or actual roles that reside in Identity Server.

The Agent provides the ability to the container to enforce such a runtime linkage of abstract security roles to real life principals. Once the Agent is installed and configured, the Application security roles can be mapped to real principals. For example, the role “Buyer” may be mapped to a Identity Server role called “Staff”. Thus when a user “Arvind” tries to access the application's protected resources, the Agent will allow this access if and only if the actual user “Arvind” is a member of the mapped role “Staff”.

An Intranet Employee Portal

An intranet employee portal may offer services such as payroll information and online benefits administration. While such services may be offered in a read-only manner to regular employees, administrators may have special privileges that can allow them to update the associated data. For instance, there could be a specialized Enterprise JavaBeans component that provides two services—one for reading payroll information and the other for updating payroll information. Using the Agent to protect this application, it will be possible to grant the administrators the privileges necessary to update payroll information, while the employees may only have read-only access.

A Content-Based Web Application

A content based web application can offer pay per-view services. The application may be partitioned into two domains—the public domain which is accessible to anonymous users, and the private domain which is accessible only to the subscribers of this particular service. Using the Agent, it will be possible to enforce that only authenticated and authorized users may be allowed to access the private domain of the application, while any user has the ability to access the public domain. Specific Servlets and JSPs that provide application functionality will be protected by the Agent by enabling the mapping of the associated security roles with actual Identity Server principals.

Supported Servers

Sun ONE Identity Server Policy Agent supports the following Application Servers:

- WebLogic 6.1 SP2 on Solaris 8, Windows 2000 Server and HP-UX 11 operating systems
- WebSphere 4.0.4 AE on Solaris 8
- Sun ONE Application Server 7.0 on Solaris 8, Solaris 9 and Windows 2000 Server.

Policy Agent for WebLogic 6.1 SP2

This chapter describes how to install and configure Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 Application Server. Topics include:

- Supported Platforms
- Guidelines
- Installing the Agent
- WebLogic Server Configuration
- Application Configuration
- Agent Configuration
- Using Agent and Sun ONE Identity Server SDK APIs
- Uninstalling the Agent

Supported Platforms

The Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 is supported on the following platforms:

- Solaris 8
- Windows 2000 Server
- HP-UX 11

How the Policy Agent for WebLogic 6.1 SP2 Works

The Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 consists of two main components that affect the operation of WebLogic 6.1 SP2 as well as the behavior of the protected application. These components are:

- **Agent Realm.** The Agent Realm component provides the ability to WebLogic 6.1 SP2 to interact with Identity Server's user and role information. This acts as the core of the Agent and has to be configured correctly in order for the Agent to function.
- **Agent Filter.** The Agent Filter component provides the ability to the hosted application to enforce Sun ONE Identity Server based authentication and is responsible for creating the Security Principal associated with the logged on user. Every application that has to be protected by the Agent must have its Deployment Descriptors changed to reflect that it is configured to use the Agent Filter component. Applications that do not have this setting will not be protected by the Agent and may malfunction or become unusable if deployed on an Application Server where the Agent Realm component is installed.

Together, the Agent Realm and Agent Filter components work in tandem with Identity Server and enforce authentication and authorization for clients trying to access protected J2EE Applications.

Guidelines

In order to use the Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 in the most optimal manner, it is recommended that you follow the following guidelines:

- **Use Agent-Based Authentication**

After the Agent is installed and the application has been configured to use Agent Filter component, the Agent Filter component enforces authentication for all web based access to the protected application's enforced portions. Working in tandem with the Agent Realm component, the Agent Filter ensures that the J2EE policies defined for the protected application get evaluated correctly based on the set role-to-principal mappings, at the same time offering other key services such as Single Sign-On (SSO). Therefore, it is recommended that protected applications do not use their own authentication mechanism or any container based authentication mechanisms which would result in the Agent Filter component being bypassed during the application operation.

- **Create Enhanced Security Aware Applications**

The Agent provides the rich APIs offered by Identity Server SDK libraries, which are available for use within the protected application. Using these APIs, the application architect can create enhanced security aware applications that are custom tailored to work in the security framework offered by Identity Server. For more information on how to use the Sun ONE Identity Server SDK, refer to *Sun ONE Identity Server Programmer's Guide*.

Installing the Agent

The Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 may be installed on platforms—Solaris 8, Windows 2000 Server, or HP-UX 11. The installation program for Identity Server Policy Agent for WebLogic Server 6.1 SP2 should be launched according to the following steps as applicable to the respective platform in use. Once the installation program is launched successfully, you may skip to the next section, which details the steps necessary to install the Agent.

Pre-installation Tasks

The following tasks must be performed before installing the Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2:

1. **Install the WebLogic Server 6.1 SP2.**

Refer to WebLogic Server documentation for the necessary details. When the server is installed, test the installation by using the provided sample application to ensure that the server is installed correctly.

2. **Test the deployment of application that you intend to protect.**

Before installing the Agent, it is important that you deploy and test the application that must be protected for simple functionality. Once it is established then the application can be deployed successfully, you are ready to install the Agent.

Launching the Installation Program on Solaris 8

The binaries for Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 for Solaris platform are provided as a tar-gzip archive. Copy this archive to the machine where WebLogic Server is installed and then perform the following steps to launch the installation program:

1. Login as root.
2. Unzip the binary archive using the following command:

```
# gzip -dc  
j2eeagents-2.0-domestic-us.sparc-sun-solaris2.8.tar.gz | tar xvf  
-
```

3. Set your `JAVA_HOME` environment variable to JDK version 1.3.1 or higher. If your system does not have the required version of JDK, use the JDK supplied with WebLogic Server 6.1 SP2 server. The JDK is located under:

WebLogic_Install_Dir/bea/jdk131

The installation program provides two types of interfaces—a GUI and a command line interface. In most cases, the GUI installation program can be used for installing the Agent. However, in cases when you are installing the Agent over a telnet session on a remote server and do not have windowing capabilities, then it is recommended that you use the command-line installation program for installing the Agent. You can launch this by executing the following command:

```
# ./setup -nodisplay
```

However, if you choose to use the GUI installation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI installation program window appears on the correct console.

NOTE If you choose to use the command line installation program using the `-nodisplay` option, you may skip the following step and proceed directly to the section “Installing the Agent Using GUI,” on page 145, which details out the installation procedure.

4. Launch the GUI installation program by invoking the setup script as follows:

```
# ./setup
```

- The installation program requires that you set up your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, if you have incorrectly set the `JAVA_HOME` variable, the `setup` script will prompt you for the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the JDK installation directory used for launching the installation program. Otherwise, enter a period (.) to abort the installation.

- In order that the GUI installation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. If your `DISPLAY` environment variable is not set at the time of invoking the `setup` script, the installation program will prompt you for the `DISPLAY` environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

Provide the `DISPLAY` value to the installation program by typing the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

NOTE You can also use `agent_SunOS.class` file to install the agent. You can find this file in the directory where you have untarred the binaries

Launching the Installation Program on Windows 2000 Server

The binaries for Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 for Windows platform are provided as a zip archive. Copy this archive to the machine where WebLogic Server is installed and follow the steps to launch the installation program:

1. You must have administrative privileges when you run the installation program. If you do not have administrative privileges, either log on as “Administrator” user or request such privileges to be granted to your account by the system administrator of the machine or domain as applicable.
2. Unzip the Agent binaries in a convenient location using the Zip utility. This operation results in two executable files `setup.bat` and `setup.exe`, which may be used to launch the installation program. Each of these files provide different features for launching the installation program. You may choose either of these two files depending upon your installation requirements.

NOTE	You can also use <code>agent_WINNT.class</code> file to install the agent. You can find this file in the directory where you have unzipped the binaries.
-------------	--

Using setup.bat

In order to use the `setup.bat` file to launch the installation program, you must have a JDK version 1.3.1 or higher available in your system path. This can be verified by typing the following command in a command prompt window:

```
C:\> java -version

java version 1.3.1_02

Java(TM) 2 Runtime Environment, Standard Edition (build
1.3.1_02-b02)

Java HotSpot(TM) Client VM (build 1.3.1_02-b02, mixed mode)
```

If you do not have a JDK of required version in your system path, you can use the JDK supplied with WebLogic Server 6.1 SP2 server located at:

WebLogic_Install_Dir\bea\jdk131

The `setup.bat` may be executed by typing the file name at the command prompt window in a directory where it is present, or by double clicking the file in Windows Explorer. For example, `C:\>setup.bat`

The installation program provides two types of interfaces—a GUI and a command line interface. You can launch the installation program in the GUI mode by invoking `setup.bat` file from a command prompt window as shown above or by double clicking it in Windows Explorer. The installation program may be launched in a command line mode by passing the argument `-nodisplay` to the `setup.bat` script as follows:

```
C:\>setup.bat -nodisplay
```

Using `setup.exe`

Using `setup.exe` relieves you from the task of setting up your environment path to include a valid version of JDK. This program first checks your system for the presence of a compatible JDK version and uses the one that was found. However, if no compatible version is found, this program installs the necessary runtime and uses it to launch the installation program.

You can invoke `setup.exe` either from the command prompt or by double clicking the file from Windows Explorer. Using `setup.exe` you can launch only the GUI installation program.

NOTE	Since the command-line installation program cannot be launched using <code>setup.exe</code> in cases where the command line installation program is required, it is recommended that you use <code>setup.bat</code> to launch the command line installation program.
-------------	--

Launching the Installation Program on HP-UX 11

The binaries for Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 for HP-UX 11 platform are provided as a tar-gzip archive. Copy this archive on the machine where WebLogic Server is installed. Perform the following steps to launch the installation program:

1. Login as root.

2. Unzip the binary archive using the following command:

```
# gzip -dc
j2eeagents-2.0-domestic-us.hppa1.0-hp-hpux11.00.tar.gz | tar xvf
-
```

3. Set your `JAVA_HOME` environment variable to JDK version 1.3.1 or higher. If your system does not have the required version of JDK, use the JDK supplied with WebLogic Server 6.1 SP2 server. The JDK is located under:

WebLogic_Install_Dir/bea/jdk131

4. The installation program provides two types of interfaces—a GUI and a command line interface. In most cases, the GUI installation program can be used for installing the Agent. However, in cases when you are installing the Agent over a telnet session on a remote server and do not have windowing capabilities, then it is recommended that you use the command line installation program for installing the Agent. You can launch this by executing the following command:

```
# ./setup -nodisplay
```

However, if you choose to use the GUI installation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI installation program window appears on the correct console.

NOTE If you choose to use the command line installation program using the `-nodisplay` option, you may skip the following step and proceed directly to the next section, which details out the installation procedure.

5. Launch the GUI installation program by invoking the `setup` script as follows:

```
# ./setup
```

- The installation program requires that you set up your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, in case you have incorrectly set the `JAVA_HOME` variable, the `setup` script will prompt you for the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the JDK installation directory used for launching the installation program. Otherwise, enter a period (.) to abort the installation.

- In order that the GUI installation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. In case your `DISPLAY` environment variable is not set at the time of invoking the setup script, the installation program will prompt you for the `DISPLAY` environment variable value as follows:

Please enter the value of `DISPLAY` variable (Enter "." to abort):

Provide the `DISPLAY` value to the installation program by typing in the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

Installing the Agent Using GUI

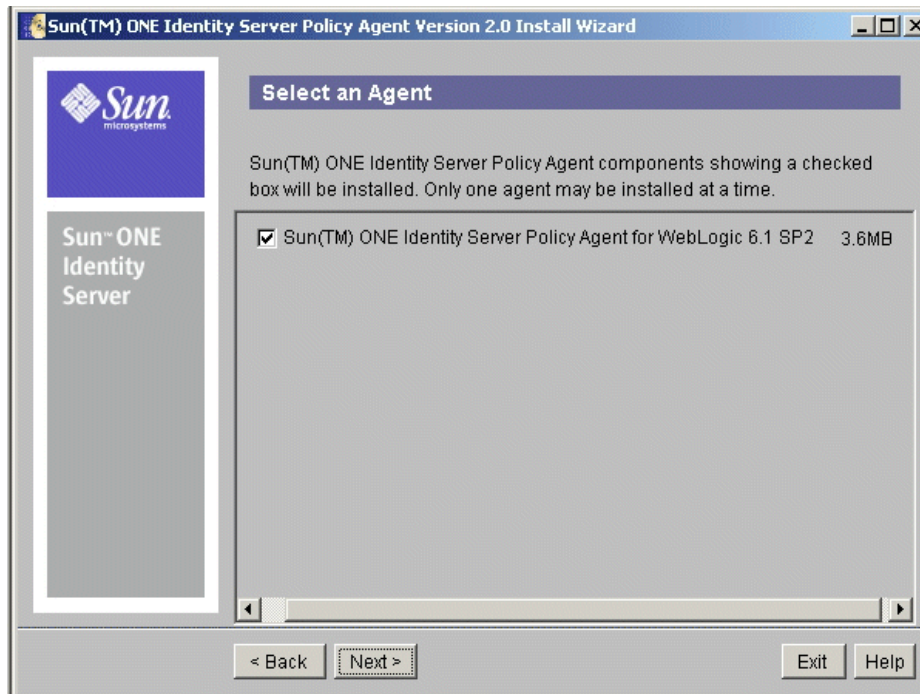
The installation program begins with a Welcome screen. Click Next to step through the installation screens, answering the questions.

1. Read the License Agreement. Click Yes (Accept License) to continue with the Installation.
2. In the Select Installation Directory screen, enter the path where you want to install.

If you wish to install the Agent in a directory different from the default directory, click the Browse button and choose the directory. Once you have selected the appropriate directory, click the Next button to proceed to the next screen.

NOTE If you select a directory that does not exist on your system, the installation program will prompt you to specify if the new directory should be created. You can either choose to create this new directory by clicking Create Directory button or select a new directory by clicking Choose another Directory button.

3. In the Select an Agent screen, select the component you wish to install in this by selecting the check box against the component name. The only component available for installation is the Identity Server Policy Agent for WebLogic Server 6.1 SP2, which is selected by default.

Figure 7-1 Component Selection Screen

NOTE For a given system, only one installation of Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 is allowed at a time. If a previous installation of the Agent has not been removed completely from the system, the check box will be disabled and no selections can be made. It is recommended that you exit the installation program by clicking the Exit button and remove the old installation completely before starting the installation again.

4. In the Sun ONE Identity Server Information screen, provide the following information about the Sun ONE Identity Server and click Next.

Figure 7-2 Sun ONE Identity Server Information Screen

Sun(TM) ONE Identity Server Information

Enter the server information where the Sun(TM) ONE Identity Server Service is installed.

Sun(TM) ONE Identity Server Host:

Sun(TM) ONE Identity Server Port:

Sun(TM) ONE Identity Server Protocol: ☒ http ☐ https

Sun(TM) ONE Identity Server Deployment URI:

amAdmin Password:

Re-enter Password:

< Back Next > Exit Help

Sun ONE Identity Server Host: Enter the fully qualified host name of the system where Sun ONE Identity Server is installed.

Sun ONE Identity Server Port: Enter the port number for the Web Server that runs Sun ONE Identity Server Services.

Sun ONE Identity Server Protocol: Select the protocol that will be used by the Agent to communicate with Sun ONE Identity Server services. This protocol may either be HTTP or HTTPS.

Sun ONE Identity Server Deployment URI: Enter the URI that should be used for accessing Sun ONE Identity Server services.

amAdmin Password: Enter the password for amAdmin user.

Re-enter Password: Re-enter the password for amAdmin user for confirmation.

NOTE The password supplied during installation is recorded by the Agent in a secure manner. However, if in the future you change this password in Sun ONE Identity Server, you will have to update the password in the Agent also. This can be done by using the `agentadmin` tool provided with the Agent. Once the Agent has been installed on the system, the `agentadmin` tool can be invoked from the location:

Agent_Install_Dir/SUNWam/wlAgent/bin/agentadmin

The `agentadmin` tool is available as `agentadmin.bat` on the Windows platform.

To change the password, invoke this tool as follows:

```
#./agentadmin -password oldpassword newpassword
```

5. In the Directory Server Information screen, provide the following information about the Directory Server that is associated with Sun ONE Identity Server services.

Figure 7-3 Directory Information Screen

The screenshot shows the 'Directory Information' screen of the Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard. The window title is 'Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard'. On the left, there is a vertical sidebar with the Sun Microsystems logo and the text 'Sun ONE Identity Server'. The main area has a header 'Directory Information' and a sub-header 'Enter the directory information corresponding to the Sun(TM) ONE Identity Server Services.' Below this, there are four input fields: 'Directory Host' with the value 'arvind.red.iplanet.com', 'Directory Port' with the value '389', 'Root Suffix' with the value 'dc=iplanet,dc=com', and 'Installation Organization' with the value 'dc=iplanet,dc=com'. At the bottom, there are four buttons: '< Back', 'Next >', 'Exit', and 'Help'.

Directory Host: Enter the fully qualified host name of the system where the Directory Server is installed.

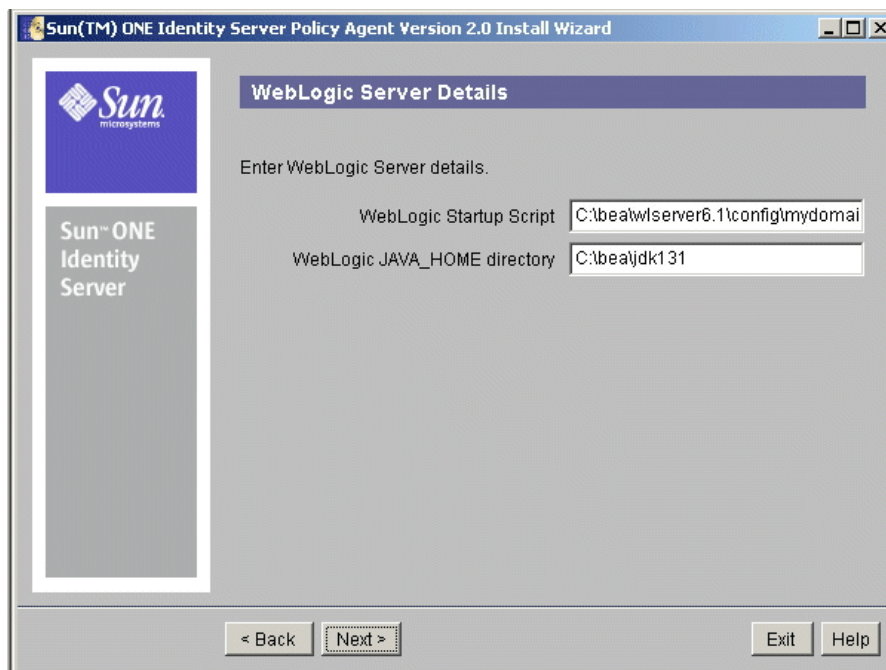
Directory Port: Enter the port number used by the Directory Server.

Root Suffix: Enter the root suffix to be used with this Directory Server.

Installation Organization: Enter the name of the installation organization as used when installing the Sun ONE Identity Server

6. In the WebLogic Server Details screen, provide the following information about the WebLogic Server on which the Agent is installed.

Figure 7-4 WebLogic Server Details Screen



WebLogic Startup Script: Enter the full path to the location of the script used to start the WebLogic Server. The WebLogic Server startup script on Solaris platform is a shell script that is used to start the WebLogic Server. On Windows platform, this is a CMD script which serves the same purpose. This script is located under the following directory:

/WebLogic_Install_Dir/bea/wlserver6.1/config/server-domain-name/

WebLogic JAVA_HOME directory: Enter the full path to the location of the JDK installation home used by the WebLogic Server. The WebLogic `JAVA_HOME` directory refers to the JDK installation that is used by the WebLogic Server. Typically, the value of this is the full path to the following directory:

/WebLogic_Install_Dir/bea/jdk131

In case you are not sure of the exact location of this directory, open the WebLogic Startup Script that is used for starting the WebLogic Server and locate the value of `JAVA_HOME` variable as specified in this file.

-
- CAUTION**
- The installation program modifies the WebLogic Server startup script to include certain libraries in the WebLogic `CLASSPATH`, as well as adds certain required parameters to the command that loads the startup classes of WebLogic Server in the Java Virtual Machine. If you specify an incorrect value for the WebLogic Server startup script, the necessary classes and parameters will not get added, resulting in malfunction of the Agent, which can render the WebLogic Server unusable. To avoid this problem, ensure that the value you specify for WebLogic Server Startup Script is accurate. See Appendix A.
 - The installation program adds certain extensions to the JDK used by the WebLogic Server, which are needed by the Agent for successful execution. These extensions are installed in the JDK directory indicated by the above mentioned WebLogic `JAVA_HOME` value. If the value supplied is incorrect or is not the JDK used by the WebLogic Server, it will result in malfunction of the Agent and can render the WebLogic Server unusable. To avoid this problem, ensure that the value you specify for WebLogic `JAVA_HOME` is accurate.
-

7. In the Agent Configuration Details screen, provide the key configuration information necessary for the Agent to function correctly.

NOTE Read the section “Important Tips” before performing this step.

Figure 7-5 Agent Configuration Details Screen

Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard

Agent Configuration Details

Enter Agent configuration values.

Agent Audit Log File: C:\Sun\SUNWam\logs\auditagent.l

Enable Audit Log File Rotation: ☒

Enable Console Integration: ☒

Host URL: http://serendipity.red.ipplanet.com:70

Login Attempt Limit: 5

Enable Not-Enforced List Cache: ☒

Number of Entries in Cache: 1000

Cache Expiration Time in Seconds: 60

Enable LDAP Attribute Headers: ☒

< Back Next > Exit Help

Audit Log File: Enter the complete path to the log file to be used by the agent to record Audit messages.

Enable Audit log file rotation: Select this to enable rotation of Audit Log files.

Enable Console Integration: Select this to enable console level integration of Sun ONE Identity Server with WebLogic Server Administration console.

Host URL: Enter a valid URL to be used as the base URL by the Agent to redirect users as necessary. This value may not be left blank and should have a valid FQDN of the Agent enabled Server. For example, if the Agent is installed on a WebLogic Server that may be accessed by using the URL `http://www.mycompany.com:80/`, then the Host URL should be set to `http://www.mycompany.com:80/` as well.

Login Attempt Limit: Enter the number of unsuccessful access attempts in succession after which the user will not be allowed to access the requested URL temporarily for security purposes. Specify the value 0 to disable this feature.

Enable Not-Enforced List Cache: Select this to enable caching of Not-Enforced List evaluation results.

Number of Entries in Cache: Specify the number of entries that the cache can hold at a given instance.

Cache Expiration Time: Specify the time (in seconds) to be used as the maximum time limit for entries that are added in the Not-Enforced List cache.

Enable LDAP Attribute Headers: Select this to enable the passing LDAP attributes associated with the current user as HTTP Headers.

Important Tips

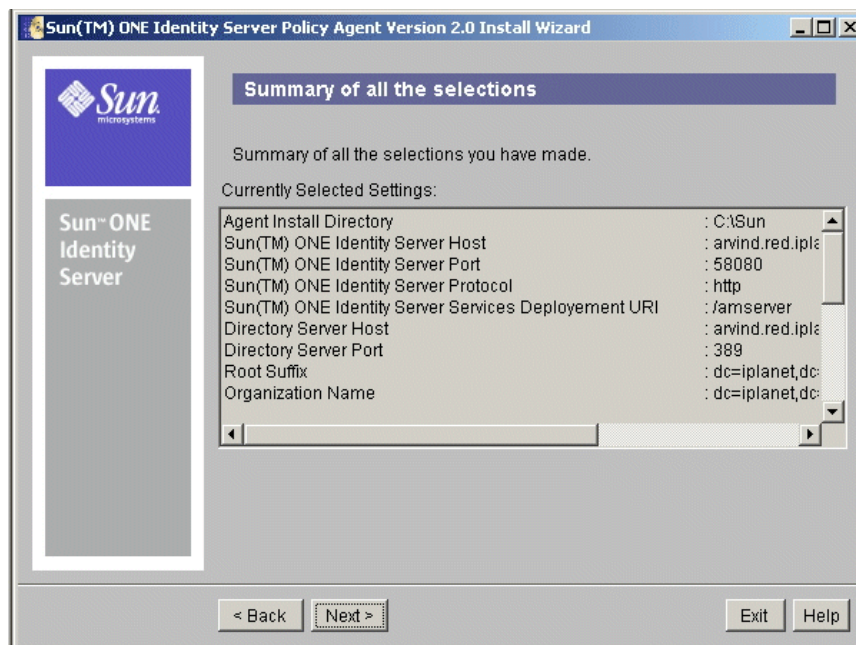
- The Audit Log file is a necessary requirement for the Agent. You may provide the name of a non-existing file on the system to be used as the Audit file. In which case, the Agent creates this file and the necessary directories in its path on first use. Alternatively, you can provide the file name of an existing file which can be used by the Agent as well. However, in either case, it is required that the specified file has write permissions for the WebLogic Server process because the Agent executes in the same process as the WebLogic Server. Providing an incorrect value for this will result in the malfunction of the Agent which can render the WebLogic Server unusable.
- Console Integration implies that the information regarding the names of all the Users, Roles, and the Users who are assigned to these Roles in Identity Server will be visible on the WebLogic Server Administration Console. Therefore, this feature must be enabled with caution since the User/Role information will then be accessible to administrators who access the WebLogic Server Console.
- When a request is intercepted by the Agent without sufficient credentials, the Agent redirects the user to the Sun ONE Identity Server's authentication service. Along with this redirect, the Agent also passes information regarding the original request to the authentication services, which is used to redirect the user back to the original requested destination. A part of this request is the Host URL that is used to identify the web container/server which the user was originally trying to access. This Host URL can be specified by setting the Host URL value on this screen. The Host URL is also used by the Agent to provide for the default FQDN to be used in cases where required. For example, if the user types in the URL `http://mycompany/SomeApp/SomeModule` and the Host URL is set to `http://www.mycompany.com:80/`, the Agent will first redirect the user to `http://www.mycompany.com:80/SomeApp/SomeModule` before taking any other action. This will ensure that the domain specific SSO Cookie that is used by the Agent to identify the user is available. Another implication of having a Host URL is that no matter which web container/server that the user was originally trying to access, the user will

be sent to the specified Host URL after successful authentication. This configuration value can also be used to override the default behavior, however, an incorrect value may lead to the application becoming inaccessible. It is therefore recommended that this value be left to the default value chosen by the installer unless there is a specific need based on the deployment scenario, which calls for setting this value appropriately.

- When Specifying the Host URL in the installer screen, ensure that the protocol, fully qualified host name, and the port that are displayed by the installer are valid and match the actual deployment scenario. For instance, if you intend to use the Agent to protect a WebLogic Server in the SSL mode, you must ensure that the Host URL has “https” as its protocol.
- The Login Attempt Limit feature can be used to guard the hosted application from Denial-Of-Service attacks where the end user can overload the application server by repeated authentication requests. By disabling this feature, the system remains vulnerable to such attacks. Therefore this feature should not be disabled unless there is a specific requirement that necessitates the disabling of this feature.
- When the Agent must process a large list of Not-Enforced pattern rules specified in the configuration, every incoming request must be evaluated against every such rule to determine if the request can be allowed without authentication or not. In scenarios where the user load is high, the time spent in evaluating these rules can add up and degrade the overall performance of the system. To avoid this problem, it is recommended that Not-Enforced List Cache be enabled.
- Enabling the Not-Enforced List Cache may result in degraded performance if the values for Number of Entries in Cache and Cache Expiration time are not set up appropriately. In case when the cache expiration time duration is more than necessary, the cache will get filled up very fast and new requests will still be evaluated against all the specified pattern rules, resulting in no improvement in performance of the system. If the Number of Entries in Cache is set very high, it may result in excessive consumption of system memory, thereby leading to degraded performance. Therefore, these values must be set up after careful deliberation of the deployment scenario and should be changed as necessary to reflect changing usage scenario of the system. It is recommended that the system be tested with various values of these two parameters in a controlled environment to identify the optimal values, and only then be deployed in production.

- The Agent maintains two caches in its memory—one for recording the URIs that were evaluated as enforced and the other for recording the URIs that were evaluated as not-enforced. The specified values of Number of Entries in Cache and Cache Expiration time is equally applicable to both of these caches. This factor must be considered when setting the values for the size and expiration time of the cache.
 - By enabling the LDAP Attribute Headers, for every incoming request, the Agent must retrieve the LDAP Attributes associated with the authenticated user and add them as Header values in the request. This feature should be used only in the case when the deployed application requires these header values for business logic implementation. Turning this feature on without an appropriate need will result in performance degradation of the system that can be otherwise avoided and serves no purpose.
 - The parameters entered during installation can be modified later by editing the `AMAgent.properties` file, see “Agent Configuration.”
8. In the Summary of all the selections screen, review and verify the installation options that you have specified for the Agent. If you need to make changes, click Back. Otherwise, click Next to proceed.

Figure 7-6 Summary of all the selections Screen



9. In the Ready to Install screen, click Install Now button to begin the installation.
10. The Install Progress screen displays the progress of the installation as the installation program makes changes to your system. If necessary, you may interrupt this process by clicking the Stop button.

NOTE It is strongly recommended that you do not interrupt this process as that may lead to a partially installed product, which may also cause problems with uninstall, and may as well render the WebLogic Server unusable. This process should be interrupted only in cases where it is absolutely necessary.

11. In the Installation Summary screen, click Details for a detailed summary of the configuration information that was processed during installation. Click Exit to end the program.

NOTE If the status of the installation is “Failed”, you must view the install log file by clicking the Details button to identify the unsuccessful installation task. In this situation, you may uninstall the Agent and try again after fixing the root cause of failure during this installation.

After the agent is installed, the next step is to configure the WebLogic Server and the deployed application appropriately as explained in the following sections.

WebLogic Server Configuration

Once the Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 has been installed on your system, the WebLogic Server must be configured to use the Agent Realm provided as a part of the Agent.

Installing the Agent Realm

The Agent Realm is a Custom Security Realm that can be added to the WebLogic Server by using the WebLogic Server Administration Console. This section outlines the steps necessary to successfully add the Agent Realm to the WebLogic Server. It must be noted that the information provided in this section is only to facilitate the

installation of Agent Realm and should not be taken as a substitute for the information provided in WebLogic Server documents. For a complete in-depth discussion on WebLogic Custom Realms, refer WebLogic Server documentation at: <http://www.bea.com>.

In order to install the Agent Realm, the following steps must be performed:

1. Create a Custom Realm for Agent
2. Create a Caching Realm for Agent Realm
3. Configure the File Realm

Create a Custom Realm for Agent

The following steps outline how a new Custom Realm may be created for installing the Agent Realm in WebLogic Server:

1. Logon to the WebLogic Server Administration Console. Use the configured system username and password for logging on to the console.
2. In the left pane of the WebLogic Server Administration Console, expand the Security Node by clicking the “+” sign.
3. In the left pane under Security node, click the “Realms” item to display a list of available Realms in the system on the right pane.
4. On the right pane, click the “Configure a new Custom Realm” link. This displays a form that can be used to enter the information regarding the new Custom Realm that you are trying to create.
5. In this form, enter the following information and click on Create:

Name: Agent Realm

Realm Class Name: `com.ipplanet.amagent.weblogic.realm.AgentRealm`

6. After creating the new Realm, restart the WebLogic Server.

Once the WebLogic Server is restarted, using the Administration Console, navigate to Security > Realms node. You should be able to see the newly created Agent Realm in the list of Realms displayed on the right pane.

Create a Caching Realm for Agent Realm

Follow these steps to create a new Caching Realm for installing the Agent Realm in WebLogic Server:

1. Logon to the WebLogic Server Administration Console. Use the configured system username and password for logging on to the console.

2. In the left pane of the WebLogic Server Administration Console, expand the Security Node by clicking the “+” sign next to it.
3. In the left pane under Security node, click the Caching Realms to display a list of available Caching Realms in the system on the right pane.
4. On the right pane, click the “Configure a new Caching Realm” link. This displays a form that can be used to enter the information regarding the new Caching Realm that you are trying to create.
5. In this form, enter the following information:
 - Name:** Agent Caching Realm
 - Basic Realm:** Select Agent Realm from the pull down menu.
6. Click on the Create button. This refreshes the right pane and a new Caching Realm is created. The right pane displays the configuration of this newly created Caching Realm.
7. In the right pane, disable all the caching attributes. This can be done as follows:
 - c. Click the ACL Tab. This displays the ACL caching attributes.
 - d. Uncheck the check box next to Enable ACL Cache.
 - e. Click on the Apply button.
 - f. Repeat this process for the rest of the available tabs—Authentication, Groups, Users, and Permissions. Uncheck the appropriate Enable Cache check box and click the Apply button.
8. Restart the WebLogic Server.

Once the WebLogic Server is restarted using the Administration Console, navigate to Security > Caching Realms node. You should be able to see the newly created Agent Caching Realm in the list of Caching Realms displayed on the right pane.

Configure the File Realm

Once the Agent Caching Realm has been created, the WebLogic Server must be configured to use this new Caching Realm. This is done by configuring the File Realm. The following steps outline how the File Realm may be configured for this purpose:

1. Logon to the WebLogic Server Administration Console.
2. On the left pane click the Security node. This forces the console to display the Security configuration of the WebLogic Server in the right pane.

3. In the right pane, click the Filerealm tab. This causes the console to display the details of the current File Realm.
4. In the form that is displayed on the right pane, under Caching Realm, select the Agent Caching Realm from the pull down menu.
5. Click the Apply button.
6. Restart the WebLogic Server.

Once the File Realm is configured and the WebLogic Server is restarted, the Agent Realm has been successfully installed.

NOTE	Once the Agent Realm has been successfully configured, it is recommended that you create a backup of the WebLogic Server's <code>config.xml</code> file. You may name this backup as <code>config.xml.withAgent</code> , so that the next time you install the Agent, you can simply copy this file on top of your existing <code>config.xml</code> and bypass the manual steps necessary to install the Agent Realm.
-------------	---

Troubleshooting the Installation

If after configuring the File Realm, the WebLogic Server does not start up correctly, the following could be the reasons:

- **The WebLogic Server Startup File was not modified successfully by the Agent Installation.**

This can happen if you have modified the WebLogic Server startup file before installing the Agent, or if the installation program did not have sufficient permissions to modify this file. In either case, refer to the information provided in Appendix A to manually modify the startup file in order to recover from this problem.

- **The Agent installation program was unable to install the required extensions for the JDK being used by WebLogic Server.**

To verify this, first look at the WebLogic Server startup file to determine the exact location of JDK used by the WebLogic Server. This can be inferred from the value of `JAVA_HOME` as set in the WebLogic Server startup file. Using this value as the JDK directory, ensure that it has the necessary extensions installed in it. Refer Appendix A to manually install the extensions if necessary in order to recover from this problem.

If the above reasons do not explain why the WebLogic Server did not startup correctly, it implies that there has been a serious installation error, which has resulted in the WebLogic Server being unusable. To recover from this problem, uninstall the Agent from your system. This should bring the WebLogic Server back to the state that it was before the installing the Agent.

Application Configuration

The Agent Realm component of Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 provides runtime mapping of various principals in Sun ONE Identity Server. Abstract security role names are used by the hosted application in order to determine if the currently authenticated user is authorized to access a particular resource or is otherwise a member of a given role. This runtime evaluation can occur only if the user is authenticated as an Identity Server principal by the means of Identity Server's authentication service. Without the user being authenticated appropriately, the results of such evaluations done by the Agent Realm will always be negative, resulting in access being denied to the user for the requested resource.

It is the Agent Filter component that enforces authentication for users who try to access particular application resources, thereby enabling the Agent Realm component to correctly evaluate the principal mappings as desired.

Unlike the Agent Realm component which is installed in the core of WebLogic Server, the Agent Filter is installed in the deployed application which must be protected by Identity Server. This is true for every application that must be protected on the WebLogic Server using the Agent. It is recommended that applications that are not protected using the Agent should not be deployed on the WebLogic Server on which the Agent Realm has been installed. This is to ensure that such applications can independently enforce their own security requirements as necessary. The presence of Agent Realm will interfere with the security evaluations done by such applications resulting in their malfunction.

Installing the Agent Filter Component in an Application

The Agent Filter can be installed by simply modifying the deployment descriptor of the application that needs to be protected. The following steps outline the process to install the Agent Filter component for a given application:

1. If the application is currently deployed on the WebLogic Server, it must be removed using the WebLogic Server's Administration Console or by the use of WebLogic Server's deployment tools.
2. It is recommended that you create a backup of the deployment descriptor that will be edited in order to install the Agent Filter in this application.
3. Edit the application's `web.xml` deployment descriptor. Since the Filters were introduced in Servlet Specification 2.3, the `web.xml`'s DOCTYPE element must be changed to reflect that the deployment descriptor is a Servlet 2.3 compliant deployment descriptor. This can be done by setting the DOCTYPE element as:

```
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
```

4. Once the DOCTYPE element has been changed, add the Filter elements in the deployment descriptor. This can be done by specifying the Filter element and the Filter-mapping element in the `web.xml` deployment descriptor immediately following the description element of the `web-app` element. The following is a sample `web.xml` with the Filter and Filter-mapping elements.

```
<web-app>
  <display-name>...</display-name>
  <description>...</description>

  <filter>
    <filter-name>Agent</filter-name>
    <display-name>Agent</display-name>
    <description>Sun™ ONE Identity Server Policy Agent for
WebLogic 6.1 SP2</description>
    <filter-class>com.ipplanet.amagent.weblogic.filter.AgentFilter<
  /filter-class>
  </filter>
  <filter-mapping>
    <filter-name>Agent</filter-name>
    <url-pattern>*/</url-pattern>
  </filter-mapping>
  ...
  ...
</web-app>
```

5. Once the `web.xml` deployment descriptor has been modified to reflect the new DOCTYPE and filter elements, the Agent Filter has been added to the application.

Creating Role-to-Principal Mappings

Once the application has been configured to have the Agent Filter component in it, the Agent Filter will enforce authentication, thereby enabling the Agent Realm to successfully resolve the role-to-principal mappings. However, these mappings must first be created in order that the hosted application may use them during runtime.

The following are two ways to create such mappings:

- Editing the WebLogic Server specific deployment descriptors.

The WebLogic Server specific deployment descriptors may be edited to create the role to principal mappings. These descriptors are in `weblogic.xml` and `weblogic-ejb-jar.xml` files. Refer WebLogic Server reference documentation to learn the details of how these descriptors may be edited to create the role to principal mappings. Alternatively, you can refer to the information provided in Appendix B for sample descriptors, which create such mappings.

- Using the WebLogic Server Administration Console.

The WebLogic Server administration console allows you to create the role to principal mappings by editing the deployed application's deployment descriptors on the fly. In order to use this facility, the application must be deployed and the WebLogic Server must be up and running at that time. Refer to the WebLogic Server documentation on how to use the Administration console to create such mappings for deployed applications.

Application Specific Agent Configuration

Often, the deployed applications are partitioned into public and protected parts, which have varying access restrictions. In most cases, the public portions of the application are accessible to anonymous users, whereas the protected portions of the application are accessible only to the registered users. The Agent can be configured to allow this type of access by letting anonymous users access the public portions of the application without requiring that they authenticate using Identity Server's authentication service. This information is provided as a part of the Agent's configuration properties file that is present in the following location:

Agent_Install_Dir/wlAgent/amAgent/config/AMAgent.properties

This file may be edited to provide general as well as application-specific configuration for the Agent.

NOTE

- The properties specified in the `AMAgent.properties` file are required for the Agent to function properly. Invalid values specified in this file can lead to malfunction of the Agent, application becoming inaccessible, or the entire system to become unusable. It is recommended that you use extreme caution when modifying the values in this file and always create backups before such modifications to ensure that you can back-out your changes to restore the system to its original state.
- The properties specified in the `AMAgent.properties` file are loaded during the WebLogic Server startup time. Any changes made to this file while the WebLogic Server is running will not take effect until the server has been restarted.

Providing Application-Specific Not-Enforced List

In order to allow anonymous users to access portions of the application, the `AMAgent.properties` file must include an entry for the specific application. The entry that specifies this property is called the application not-enforced list and is identified by the string:

```
com.sun.am.policy.config.filter.AppName.notEnforcedList[index]=pattern
```

This property requires that you format it correctly in order that it can be used by the Agent during runtime. The entries appearing in italics within this string should be replaced by their appropriate values as follows:

AppName: This string should be replaced by the deployed application's context path without its leading forward-slash (/) character. The context path is the first URI segment that is used to identify which application the user is trying to access. For example, if the user accesses your application by typing the URL

```
http://myserver.mydomain.com/SomeApp/index.html, or
```

```
http://myserver.mydomain.com/SomeApp/SomeModule/doSomething.jsp
```

then, in both these cases, the *AppName* is `SomeApp`.

index: This is an integer value starting from 0 for every deployed application and is unique for every entry in the application not-enforced list. For example, the following are two entries of application not-enforced list for an application with context path `/SomeApp`:

```
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[0]=/SomeApp/public/*
```

```
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[1]=/SomeApp/images/*
```

pattern: This is a pattern string that will be matched with the incoming request to evaluate if the request should be allowed to pass without enforcing authentication or not. The pattern string could be a specific URI, for example `/SomeApp/public/RegistrationServlet`, or could be a generic pattern using the wild card character '*' that can be used for denoting 0 or more characters in the request URI; for example `/SomeApp/public/*` will match with any URI that begins with `/SomeApp/public/`.

Using this property, you could specify none or many pattern strings and URIs that the Agent will treat as not-enforced. In other words, user requests that match these particular patterns will be allowed to pass through without enforcing authentication.

Providing Application Specific Access Denied URI

In cases when the Login Attempt Limit is enabled (see "Agent Configuration" for information on how this feature is configured), the Agent is required to block the user's access under certain circumstances. The default behavior of the Agent in this situation is to send an HTTP Status Code 403 Forbidden. In such a situation, the web container can display its preconfigured Forbidden page or simply send the status code in which case the user's browser displays the details of the error message in its own manner. While this is the default behavior of the Agent, it can be changed to suit the needs of the application by allowing the Agent to use an application-specific URI that will be used as the access denied error page.

This can be done by setting the following property in `AMAgent.properties` file:

```
com.sun.am.policy.config.filter.AppName.accessDeniedURI=URI to use
```

This property requires that you format it correctly in order that it can be used by the Agent during runtime. The entries appearing in italics within this string should be replaced by their appropriate values as follows:

AppName: This string should be replaced by the deployed application's context path without the leading forward-slash character (/). The context path is the first URI segment that is used to identify which application the user is trying to access. For example, if the user accesses your application by typing the URL

```
http://myserver.mydomain.com/SomeApp/index.html, or
```

```
http://myserver.mydomain.com/SomeApp/SomeModule/doSomething.jsp
```

then, in both these cases, the *AppName* is `SomeApp`.

URI to use: is an application specific URI that the Agent will use to locate the display page for blocking the user request. This URI can be a static HTML page, or a JSP or even a Servlet. However, this URI must be a part of the application itself. In other words, this URI must begin with:

/AppName/rest of the URI

Special Case: Default Web Application

A default web application in WebLogic Server is accessible without providing any context path in the request URI. For example, the following URL is that of a Default web application:

```
http://myserver.mydomain.com/index.html
```

Clearly, this URL does not have an associated context path.

For such applications, the Agent provides a convenient means of identifying that an entry is specific to the default web application. This is done in two steps as follows:

1. The following property is set to a name that represents the default web application:

```
com.sun.am.policy.config.filter.defaultWebAppName= DefaultWebApp
```

2. This name is then used to specify the application not-enforced list as well as the application's access denied URI as follows:

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[0]
=/index.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]
=/about.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.accessDeniedURI=/U
RLAccessDenied.html
```

Using this scheme, the default web application that does not have a context path associated with it, may be configured just like any other application that has a context path. The same rules apply to the default web application for specifying the not-enforced list entries and access-denied URI as are applicable for the rest of the applications. However, the only difference is that the access-denied URI of the default web application as well as the not-enforced list entries cannot begin with the `/DefaultWebApp/` path segment since such a path segment does not exist on the application server in reality. The *AppName* in this case where the actual context path is an empty string, is only provided as a convenience to specify the properties associated with the default web application and should not be used in specifying their values.

Global Agent Configuration

The `AMAgent.properties` file provides a way to specify a global not-enforced list, which will be applicable to all the protected applications that are deployed on the server. This list is specified by using the following property:

```
com.sun.am.policy.config.filter.global.notEnforcedList[index]= pattern
```

pattern is either an exact URI or a pattern specified by using the wild card character '*' that can be substituted for zero or more characters in the request URI.

index is an integer value starting from 0 and unique for every entry.

Not-Enforced List Usage Considerations

Although the use of Not-Enforced List can be extremely helpful in partitioning your application for public and protected domains, it can also lead to undesirable effects if not used appropriately.

For example, if a request URI that represents a Servlet is matched by some not-enforced list pattern, then the Agent Filter will not enforce authentication for users who try to access that particular Servlet. However, consider the case where this Servlet accesses an Enterprise JavaBeans component that is protected by the Agent using role to principal mapping. In such a case, since the user is not authenticated, the access to the protected component will result in a security violation exception being generated by the application server. Therefore, before an entry is added to the not-enforced list, it must be ensured that it does not in any way cover a resource that may be protected or may try to access a protected resource.

Another interesting aspect of the use of non-enforced list are the images. Typically in a web page there are many images for various purposes like buttons, place holders, banners, and logos. Every time the user accesses this page, the browser issues a request to the application server to get the images contained in this page. Each of such requests are treated as individual requests coming from the client and goes through the same evaluation mechanism for authentication and not-enforced list check as does any other request. This results in one client generating multiple calls to the server for displaying a single page. Considering the overhead involved in enforcing authentication for every such request, it can impact the overall performance of the system. A solution to this problem is to have a global not-enforced list entry or entries that match all images.

For example:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=*.gif
```

```
com.sun.am.policy.config.filter.global.notEnforcedList[1]=  
/images/*
```

This indicates that any request URI that ends with `.gif` will be not enforced and nor will be any URI that begins with `/images/`. In heavy user load situations, this can significantly increase the performance of the system.

Agent Configuration

The core configuration needed by the Sun ONE Identity Server Policy Agent for WebLogic Server 6.1 SP2 is provided in the `AMAgent.properties` file located in the following directory:

Agent_Install_Dir/wlAgent/amAgent/config

This property file provides many configuration settings that can be modified in order to customize the Agent's operation for your deployment scenario.

NOTE	Before proceeding, it is important to note that this file and the information within it are critical for the operation of the Agent. It is strongly recommended that you always create backup of this file before modifying it. Also it is strongly recommended that you do not modify this file unless it is absolutely necessary. Note that invalid data entries present in this file can lead to the malfunction of the Agent, malfunction of the deployed applications, and could render the entire system unusable.
-------------	--

The settings provided in this file can be classified into the following categories:

- Common Configuration
- Audit Configuration
- Realm Configuration
- Global Filter Configuration
- Application Filter Configuration
- Debug Engine Configuration

Following sections detail the settings for each of these classifications.

Common Configuration

Settings in this section are general settings that affect the behavior of the Agent as a whole.

Organization Name

Key: `com.sun.am.policy.config.org`

Description: This property specifies the organization name to be used when searching for principals in Sun ONE Identity Server.

Valid Values: String representing the organization name in Sun ONE Identity Server. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example: `com.sun.am.policy.config.org=iplanet.com`

Root Suffix

Key: `com.sun.am.policy.config.rootsuffix`

Description: This property specifies the root suffix to be used when searching for principals in Sun ONE Identity Server

Valid Values: String representing the root suffix in Sun ONE Identity Server. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example: `com.sun.am.policy.config.rootsuffix=o=isp`

People Container Level

Key: `com.sun.am.policy.config.realm.peopleContainerLevel`

Description: This property specifies the people container level to be used when searching for principals in Sun ONE Identity Server.

Valid Values: Non-zero unsigned integer representing the People Container Level in Sun ONE Identity Server, which may be used when searching for principals. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example: `com.sun.am.policy.config.realm.peopleContainerLevel=1`

Audit Configuration

These settings are exclusively used to configure the Audit Engine used by the Agent.

Language Code

Key: `com.sun.am.policy.config.audit.localeLanguageCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeLanguageCode` must be a valid ISO Language Code. Default value of this property is `en`

NOTE For more information, refer ISO 639 specification at <http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt>

Example: `com.sun.am.policy.config.audit.localeLanguageCode=en`

Country Code

Key: `com.sun.am.policy.config.audit.localeCountryCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeCountryCode` must be a valid ISO Country Code. The default value of this property is `US`

NOTE For more information, refer ISO 3166 specification: http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html

Example: `com.sun.am.policy.config.audit.localeCountryCode=US`

Audit Log File

Key: `com.sun.am.policy.config.audit.logfile.name`

Description: This property specifies the Audit log file to be used for recording Audit messages.

Valid Values: String representing the complete path name of the file to be used by Agent to record Audit messages.

NOTE

- Be sure that the WebLogic Server process has sufficient permissions to write to this file.
- Invalid value specified for this property may result in the failure of the system to start up correctly.

Example:

```
com.sun.am.policy.config.audit.logfile.name=/audit/agent.log
```

Audit Log File Rotation Flag

Key: `com.sun.am.policy.config.audit.logfile.rotate`

Description: This property specifies if the Audit log file should be rotated by the Agent.

Valid Values: `true/false`. The default value of this property is `false` and should be changed as necessary.

Example: `com.sun.am.policy.config.audit.logfile.rotate=false`

Audit Log File Rotation Size

Key: `com.sun.am.policy.config.audit.logfile.rotate.size`

Description: This property specifies the approximate size of the Audit log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated. The default value of this property is 52428800 bytes (~ 50 MB) and should be changed as required.

Example: `com.sun.am.policy.config.audit.logfile.rotate.size=52428800`

NOTE

This property is not used when audit log file flag value is `false`.

Realm Configuration

These settings are used to configure the Agent Realm component.

Allow Console Integration Flag

Key: `com.sun.am.policy.config.weblogic.allowConsoleIntegration`

Description: This property specifies if the Agent Realm should allow console level integration of Sun ONE Identity Server with WebLogic Server.

Valid Values: true/false

-
- NOTE**
- When set, the result of console level integration is that the WebLogic Server Administrator will be able to see the list of Sun ONE Identity Server principals in WebLogic Server Console.
 - Since this setting enables the WebLogic Server Administrator to see Sun ONE Identity Server principals in WebLogic Server console, it should be used with caution.
 - The default value of this setting is false and should be changed as necessary.
-

Example:

```
com.sun.am.policy.config.weblogic.allowConsoleIntegration=true
```

SSO Cache Cleanup Size

Key: com.sun.am.policy.config.realm.ssoCacheCleanupSize

Description: This property specifies the maximum number of entries in the SSO Cache maintained by the Agent Realm after which a cleanup will be initiated.

Valid Values: Any positive integer value indicating the number of entries in the Agent Realm's SSO Cache which when exceeded will initiate a clean up operation to ensure minimal and appropriate use of the system's memory.

-
- NOTE**
- Values that are valid but not suited for the deployment scenario may result in degradation of system performance and/or result in the loss of availability of SSO token in the system.
 - The value for optimal system performance will depend on the type of application deployed, the number of active user sessions that the application is subject to during peak periods, and the average user session time value.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 1000.
-

Example: com.sun.am.policy.config.realm.ssoCacheCleanupSize = 1000

SSO Cache Cleanup Lock Time

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime`

Description: This property specifies the amount of time in seconds that the Agent Realm will wait before initiating the next cleanup process for the SSO cache if the last cleanup process did not result in freeing any memory.

Valid Values: Any positive integer value indicating the number of seconds that the cleanup process will not be restarted if the last cleanup process did not result in freeing sufficient memory.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
 - The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 1200.
-

Example: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime = 1200`

SSO Cache Cleanup Bound Size

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize`

Description: This property specifies the maximum number of entries in the Agent Realm SSO Cache that will be inspected during cleanup process. This value when set appropriately will result in overall improvement of the response time of the system when it undergoes a cache cleanup operation.

Valid Values: Any positive integer value indicating the number of entries the cleanup process will inspect in the SSO cache during the cleanup operation. This value can be set independently with respect to the value of the property `com.sun.am.policy.config.realm.ssoCacheCleanupSize`.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
- The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
- To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
- This property defaults to a value of 50.

Example:

```
com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize = 50
```

Global Filter Configuration

These settings are used to configure the Agent Filter component.

SSO Token Name

Key: `com.sun.am.policy.config.filter.ssoTokenName`

Description: This property specifies the name of the Cookie that represents SSO Token.

Valid Values: A String that represents the name of SSO Token Cookie issued by Sun ONE Identity Server authentication service. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example:

```
com.sun.am.policy.config.filter.ssoTokenName=iPlanetDirectoryPro
```

FQDN Map

Key:

```
com.sun.am.policy.config.filter[invalid-name]
```

Description: The FQDN Map is a simple map that enables the Agent to take corrective action in the case where the users may have typed in an incorrect URL such as by specifying partial hostname or using an IP address to access protected resources.

Valid Values: Valid values must comply with the syntax of this property which represent invalid FQDN values mapped to their corresponding valid counterparts.

The format for specifying this property is as follows:

```
com.sun.am.policy.config.filter.fqdnMap[invalid-name]=valid-name
```

Where *invalid-name* is a possible invalid FQDN host name that may be used by the user, and the *valid-name* is the FQDN host name the filter will redirect the user to.

NOTE

- Ensure that there are no overlapping values for the same invalid FQDN name. Failing to do so may lead to the application becoming inaccessible.
- Invalid value for this property can result in the application becoming inaccessible.
- This property can be used for creating a mapping for more than one host name. This may be the case when the applications hosted on this server are accessible by more than one host name. However, the use of this feature must be done with caution as it can lead to the applications becoming inaccessible.
- The Agent gives precedence to the entries defined in this property over the `com.sun.am.policy.config.filter.hostURL` property.
- This property may be used to configure the Agent to not take corrective action for certain hostname URLs. For example, if it is required that no corrective action such as a redirect be used for users who access the application resources by using the raw IP address, you can implement this by specifying a map entry such as:

```
com.sun.am.policy.config.filter[IP]=IP
```

- Any number of such properties may be specified as long as they are valid properties conforming to the above stated requirements.
-

Example:

```
com.sun.am.policy.config.filter[myserver]=myserver.mydomain.com
com.sun.am.policy.config.filter[myserver.mydomain]=myserver.mydomain.com
```

```
com.sun.am.policy.config.filter[IP]=myserver.mydomain.com
```

```
com.sun.am.policy.config.filter[invalid-name]=valid-name
```

Login URL

Key: `com.sun.am.policy.config.filter.loginURL`

Description: This property specifies the login URL to be used by the Agent to redirect incoming users without sufficient credentials to the Sun ONE Identity Server authentication service.

Valid Values: A string that represents the complete URL to be used as the redirect URL in order to send users without sufficient credentials to Sun ONE Identity Server authentication service. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example:

```
com.sun.am.policy.config.filter.loginURL=http://myserver.mydomain.com:58080/amserver/login
```

Host URL

Key: `com.sun.am.policy.config.filter.hostURL`

Description: This property specifies the host URL to be used by the Agent to reconstruct the request issued by the browser. This value is used by Sun ONE Identity Server Authentication service to redirect the user back to the original request destination after successful authentication.

Valid Values: A string value that represents the host URL that the user is expected to access in order that the Agent can intercept. This value must match the format of this property. The format of this property value is as follows:

protocol : // *hostname* . *optional-sub-domain* . *domain* : *port*

protocol can be http or https.

hostname.optional-sub-domain.domain is the fully qualified host name that the user is expected to access in order that the Agent may intercept.

port is the port number on which the receiving web server is listening.

If left unspecified, the Agent will try to reconstruct the host URL from the request.

Example:

```
com.sun.am.policy.config.filter.hostURL=http://www.iplanet.com:80
```

Goto Parameter

Key: `com.sun.am.policy.config.filter.gotoParameter`

Description: This property specifies the goto parameter name to be used by the Agent when redirecting the user to the appropriate authentication service. The value of this parameter is used by the authentication service to redirect the user to the original requested destination.

Valid Values: A string value that represents the goto parameter name as expected by the authentication service.

Example: `com.sun.am.policy.config.filter.gotoParameter=goto`

Login Attempt Limit

Key: `com.sun.am.policy.config.filter.loginAttemptLimit`

Description: This property specifies the number of login attempts that a user can make using a single browser session.

Valid Values: Unsigned integer value, including 0, which indicates the number of login attempts allowed for any user trying to gain access to protected resources.

-
- NOTE**
- This option can be disabled by setting the value to 0
 - The default value of this property is 5
-

Example: `com.sun.am.policy.config.filter.loginAttemptLimit=5`

Login Counter Cookie Name

Key: `com.sun.am.policy.config.filter.loginCounterCookieName`

Description: This property specifies the name of the cookie that will be used to track the number of unsuccessful login attempts made by the user.

Valid Values: A String that represents the name of the cookie to be issued by Agent in order to track the number of unsuccessful login attempts made by the user. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example:

`com.sun.am.policy.config.filter.loginCounterCookieName=iPlanetLoginAttemptID`

Not-Enforced-List Cache Enable Flag

Key: `com.sun.am.policy.config.filter.notEnforcedList.cache`

Description: This property specifies if the requests URIs that are evaluated as enforced or not-enforced may be cached to increase performance of the system.

Valid Values: `true/false`. The default value of this property is `true`.

Example: `com.sun.am.policy.config.filter.notEnforcedList.cache=true`

Not-Enforced-List Cache Size

Key: `com.sun.am.policy.config.filter.notEnforcedList.cacheSize`

Description: This property specifies the number of entries that will be kept in the cache of not-enforced URIs and enforced URIs by the Agent.

Valid Values: Non-zero unsigned integer indicating the number of enforced as well as not enforced request URIs to be cached during runtime.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
 - The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries and a host of other deployment specific factors.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
-

Example:

`com.sun.am.policy.config.filter.notEnforcedList.cacheSize=1000`

Not-Enforced-List Cache Expiration Time

Key: `com.sun.am.policy.config.filter.notEnforcedList.cacheTime`

Description: This property specifies the amount of time in seconds that will be used to evaluate if a cached entry can be removed from the cache to free up resources for new cache entries.

Valid Values: Non-zero unsigned integer indicating the time in seconds that will be used as the cache expiration time for entries in the cache during cleanup operation.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
- The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries and a host of other deployment specific factors.
- To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.

Example:

```
com.sun.am.policy.config.filter.notEnforcedList.cacheTime=60
```

LDAP Attribute Header Enable Flag

Key: `com.sun.am.policy.config.filter.enableLDAPAttributeHeaders`

Description: This property specifies if the Agent should populate the `HttpServletRequest` with LDAP Attributes associated with the currently authenticated user.

Valid Values: `true/false`. The default value of this property is `false` and should be changed as necessary.

Example:

```
com.sun.am.policy.config.filter.enableLDAPAttributeHeaders=true
```

LDAP Attribute Header Map

Key: `com.sun.am.policy.config.filter.ldapAttribute[attr-name]`

Description: This property specifies the LDAP Attributes to be populated under specific header names for the currently authenticated user.

Valid Values: Valid values must comply with the syntax of this property. The specified LDAP Attribute should be a valid attribute. The specified HTTP Header name should conform to HTTP Header name conventions. The format for specifying this property is as follows:

```
com.sun.am.policy.config.filter.ldapAttribute[attr-name]=header-name
```

where *attr-name* is the name of the LDAP Attribute to be looked up for the authenticated user, and *header-name* is the name of the Header that will be used to store this value.

NOTE	<ul style="list-style-type: none">• Be sure that the specified Header Names do not conflict with existing Header names.• Any number of such properties may be specified as long as they are valid properties conforming to the above stated requirements.
-------------	--

Example:

```
com.sun.am.policy.config.filter.ldapAttribute[cn]=CUSTOM-Common-Name
com.sun.am.policy.config.filter.ldapAttribute[ou]=CUSTOM-Organization-Unit
com.sun.am.policy.config.filter.ldapAttribute[o]=CUSTOM-Organization
com.sun.am.policy.config.filter.ldapAttribute[c]=CUSTOM-Country
com.sun.am.policy.config.filter.ldapAttribute[mail]=CUSTOM-Email

com.sun.am.policy.config.filter.ldapAttribute[employeenumber]=CUSTOM-Employee-Number
```

LDAP Date Header Attribute Format String

Key: com.sun.am.policy.config.filter.ldapAttributeDateHeaderFormat

Description: This property specifies the format of Date/Time value to be expected as a result of an attribute lookup. This is required when using the specialized get methods of the javax.servlet.http.HttpServletRequest interface that return Date values for headers.

Valid Values: Valid java.text.SimpleDateFormat Time Format Syntax string. For more information, see:

<http://java.sun.com/j2se/1.3/docs/api/java/text/SimpleDateFormat.html>

The default value of this property is set to `EEE, d MMM yyyy hh:mm:ss z` and should be changed as necessary.

NOTE	Invalid value of this property may result in runtime exceptions in the application.
-------------	---

Example:

```
com.sun.am.policy.config.filter.ldapAttributeDateHeaderFormat=
EEE, d MMM yyyy hh:mm:ss z
```

Authentication Session Binding Flag

Key: `com.sun.am.policy.config.filter.authSessionBinding`

Description: This property specifies if the Agent will enforce session binding with authentication.

Valid Values: `true/false`. The default value of this property is set to `false`.

Example: `com.sun.am.policy.config.filter.authSessionBinding=false`

SSO Token URL Decode Flag

Key: `com.sun.am.policy.config.filter.urlDecodeSSOToken`

Description: This property indicates if the SSO Token needs to be URL Decoded by the Agent before it may be used.

Valid Values: `true/false`. The default value of this property is set to `true`.

NOTE Valid but inappropriate value of this property may lead to the application being unreachable by the users.

Example: `com.sun.am.policy.config.filter.urlDecodeSSOToken=true`

Default Web Application Name

Key: `com.sun.am.policy.config.filter.defaultWebAppName`

Description: This property specifies a name for the Default Web Application deployed on the application server.

Valid Values: A string consisting of lower case and or upper case letters that can be used as the name for default web application. The default value of this property is `DefaultWebApp`

NOTE This property is necessary if the protected application is deployed as the default web application.

Example:

```
com.sun.am.policy.config.filter.defaultWebAppName=DefaultWebApp
```

Global Not-Enforced List

Key: `com.sun.am.policy.config.filter.global.notEnforcedList[index]`

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters. The syntax of this property is as follows:

`com.sun.am.policy.config.filter.global.notEnforcedList[index]=pattern`

index is an integer that starts from 0 and increments for every entry in this property list

pattern is a string that represents request URIs that are not enforced by Agent.

The *pattern* string may consist of wild card character '*', which may match zero or more characters.

The *index* must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries.

NOTE No value for this property indicates empty global not-enforced list.

Example:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=*.gif
com.sun.am.policy.config.filter.global.notEnforcedList[1]= public/*
com.sun.am.policy.config.filter.global.notEnforcedList[2]=
/images/*
```

Application Filter Configuration

These settings are used to configure the Agent Filter for a particular application.

Access Denied URI

Key: `com.sun.am.policy.config.filter.AppName.accessDeniedURI`

Description: This property specifies the application-specific access denied URI for the protected application.

Valid Values: The URI within the deployed application that must be used as the access denied URI to block in coming requests when necessary.

NOTE

- This property is specific to the protected application. Therefore if there are more than one protected applications deployed on the system, there should be one property for each such application.
- This property must specify a URI that is within the application. Failure to do so can result in runtime internal server errors.
- The format for specifying this property is as follows:

```
com.sun.am.policy.config.filter.AppName.accessDeniedURI=URI
```

where *AppName* is the context path name for the deployed application and *URI* is the URI to be used. Note that the difference between *AppName* and context path of the application is that the context path has a leading / character.

- In case, the protected application is the default web application, the *AppName* should be set to the same string as specified in for the value of the property Default Web Application Name.
 - In case when this property is not specified for a given application, the Agent uses HTTP Status Code 403 (Forbidden) to indicate a blocked access.
-

Example:

```
com.sun.am.policy.config.filter.Portal.accessDeniedURI=/Portal/AccessDenied.html
```

```
com.sun.am.policy.config.filter.BankApp.accessDeniedURI=/BankApp/Block.jsp
```

```
com.sun.am.policy.config.filter.DefaultWebApp.accessDeniedURI=/URLAccessDenied.htm
```

Application Not-Enforced-List

Key: `com.sun.am.policy.config.filter.AppName.notEnforcedList[index]`

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent for a particular application.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters. The syntax of this property is as follows:

```
com.sun.am.policy.config.filter.AppName.notEnforcedList[index]= pattern
```

AppName is the context path name without the leading forward-slash character (/) for the deployed application.

index is an integer that starts from 0 and increments for every specified property for the particular application.

pattern is a string that represents the URIs that are not enforced by the Agent.

NOTE

- The *pattern* string may consist of wild card character '*', which may match zero or more characters.
 - The *index* must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries. The index value will be independent for different *AppNames* specified in this property list.
 - In case the protected application is the default web application, the *AppName* should be set to the same string as specified in for the value of the property Default Web Application Name.
 - No value for this property indicates empty not enforced list.
-

Example:

```
com.sun.am.policy.config.filter.Portal.notEnforcedList[0]=
/Portal/GuestPages/*

com.sun.am.policy.config.filter.Portal.notEnforcedList[1]=
/Portal/Registration/*

com.sun.am.policy.config.filter.Portal.notEnforcedList[2]=
/Portal/WebServices/PollServlet

com.sun.am.policy.config.filter.BankApp.notEnforcedList[0]=
/BankApp/ModuleGuestTour/*

com.sun.am.policy.config.filter.BankApp.notEnforcedList[1]=
/BankApp/index.html

com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[0]=
/index.html

com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]=
/about.html
```

Debug Engine Configuration

These settings are used to configure the Debug Engine to generate diagnostic information.

Debug Level

Key: `com.sun.am.policy.config.debug.level`

Description: This property specifies the amount of debug messages that will be emitted by the Agent's Debug Engine.

Valid Values: Any of 0, 1, 3, 7, 15, and 31. These values indicate the following:

- 0 = No debugging
- 1 = Only Error messages
- 3 = Error and Warning messages
- 7 = Error, Warning and Brief Informational messages
- 15 = Error, Warning and Verbose Informational messages
- 31 = Error, Warning and Very Verbose Informational messages

NOTE

- For better performance of the system, this property should be set to a value 0. Values other than that will affect the system performance depending upon the amount of information the Debug Engine has to emit.
 - Values other than the ones specified in the Valid Values list above will lead to invalid configuration of the Debug Engine, thereby affecting the volume of messages that will be emitted.
-

Example: `com.sun.am.policy.config.debug.level=7`

Debug Log File

Key: `com.sun.am.policy.config.debug.logfile.name`

Description: This property specifies the Debug log file to be used for recording Debug messages.

Valid Values: String representing the complete path name of the file to be used by Agent to record Debug messages.

-
- | | |
|-------------|---|
| NOTE | <ul style="list-style-type: none">• Be sure that the WebLogic Server process has sufficient permissions to be able to write to this file.• Invalid or entering incorrect path in this property will lead to Debug messages not being logged in the log file. |
|-------------|---|
-

Example:

```
com.sun.am.policy.config.debug.logfile.name=/debug/agent_debug.log
```

Debug Log File Rotation Flag

Key: `com.sun.am.policy.config.debug.logfile.rotate`

Description: This property specifies if the Debug log file should be rotated by the Agent.

Valid Values: `true/false`. The default value of this property is `false` and should be changed as necessary.

Example:

```
com.sun.am.policy.config.debug.logfile.rotate=false
```

Debug Log File Rotation Size

Key: `com.sun.am.policy.config.debug.logfile.rotate.size`

Description: This property specifies the approximate size of the Debug log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated. Default value of this property is 52428800 bytes (~ 50 MB) and should be changed as required.

-
- | | |
|-------------|--|
| NOTE | This property is not used if the Debug Log File Rotation Flag is set to <code>false</code> |
|-------------|--|
-

Example:

```
com.sun.am.policy.config.debug.logfile.rotate.size=52428800
```

Debug Time/Date Format String

Key: `com.sun.am.policy.config.debug.date.format`

Description: This property specifies the format of time stamp that is used to mark the exact time when the Debug message was recorded.

Valid Values: Valid `java.text.SimpleDateFormat` Time Format Syntax string. For more information, refer to:

<http://java.sun.com/j2se/1.3/docs/api/java/text/SimpleDateFormat.html>

NOTE

- The default value of this property is set to `<MMM d, yyyy h:mm:ss a z> 'Agent'` and should be changed as necessary.
 - Invalid value of this property will result in loss of time stamp data with debug messages.
-

Example:

```
com.sun.am.policy.config.debug.date.format=[yyyy/MM/dd HH:mm:ss zzz]
```

Debug Print STDOUT Flag

Key: `com.sun.am.policy.config.debug.print.stdout`

Description: This property specifies if the Debug Engine should print the debug messages on Standard Output stream.

Valid Values: `true/false`. The default value of this property is `true` and should be changed as necessary.

NOTE

- When set to `true`, the Debug Engine prints all debug messages on the Standard output stream. This results in the debug messages being displayed on the console window where the WebLogic Server startup script was executed from.
 - This property has no affect on the ability of Debug Engine to write to debug log files.
-

Example: `com.sun.am.policy.config.debug.print.stdout=true`

Using Agent and Sun ONE Identity Server SDK APIs

You can use the Sun ONE Identity Server SDK APIs to create security and identity aware applications. Such applications can perform custom security and identity related tasks such as application level policy enforcement by exploiting the rich security and policy infrastructure offered by the Sun ONE Identity Server. When you install the Sun ONE Identity Server Policy Agent for WebLogic Server, the Sun ONE Identity Server SDK becomes available for your application to use.

While the availability of the SDK in itself is sufficient for the developers of the application to make it security aware, the Policy Agent for WebLogic further facilitates this by ensuring that the Single Sign-On (SSO) Token is available for the logged on user who at any given point in time may be using the deployed system.

When the logged on user accesses a protected resource, the Agent Filter ensures that the user be appropriately authenticated and that the corresponding Principal be available throughout the system. The Principal instance may be accessed by the J2EE programmatic security calls such as

`HttpServletRequest.getUserPrincipal()` and `EJBContext.getCallerPrincipal()`. The Principal instance returned by these methods represent the user as authenticated by the Sun ONE Identity Server's authentication service. This Principal instance can then be used to access the user's SSO Token from anywhere within the application in the following two simple steps:

1. Downcast the Principal to the class of type:
`com.ipplanet.amagent.weblogic.realm.AgentUser`

For example:

```
.....
import java.security.Principal;
import com.ipplanet.amagent.weblogic.realm.AgentUser;
.....
Principal principal = getEJBContext().getCallerPrincipal();
AgentUser user = null;
if (principal instanceof AgentUser) {
    user = (AgentUser) principal;
}
...
```

2. Use the `AgentUser` API to retrieve the SSO Token associated with this principal. For example:

```

...
String ssoTokenId = null;
if (user != null) {
    ssoTokenId = user.getSSTokenID();
}
...

```

Once the SSO Token string is acquired, it can be used for subsequent calls into the Identity Server SDK APIs.

NOTE The Agent uses these configuration settings which ensure the availability of the SSO token associated with the user in the Agent Realm. If the values for these settings is not appropriate for your deployment scenario, it may result in the degradation of the overall system performance. See “Agent Configuration.”

Uninstalling the Agent

When you install the Sun ONE Identity Server Policy Agent for WebLogic Server software, an uninstallation program is created in the installation directory. Using this uninstallation program the Agent can be removed completely from your system. While the uninstallation program deletes all the installed files from your system, certain files such as audit log messages are not deleted. You can delete them manually.

Pre-Uninstallation Tasks

Before using the uninstaller, you must complete the following tasks in the order they are listed here. Failing to perform all or any of these tasks may result in the application becoming unusable or may result in the entire system becoming unstable and unusable.

1. **Agent Filter removal:** Remove all the protected applications from the WebLogic server and edit their deployment descriptors to remove any references to the Agent Filter that were added during Agent installation. Also, you may remove any role-to-principal mappings that were done for these

applications. Once the deployment descriptors have been edited, you may redeploy the application to the WebLogic Server. Refer WebLogic Server documentation for information on how to remove and redeploy applications that are already deployed on your system.

2. **Agent Realm removal:** The next step in order to successfully uninstall the Agent is to remove the Agent Realm configuration. This can be done by the following steps given below:
 - a. Logon to the WebLogic Server Administration Console. Use the configured system username and password for logging on to the console.
 - b. On the left pane click the Security node. This forces the console to display the Security configuration of the WebLogic Server in the right pane.
 - c. In the right pane, click on the Filerealm tab. This causes the console to display the details of the current File Realm.
 - d. In the form that is displayed on the right pane, under Caching Realm, select the appropriate Caching Realm other than the Agent Caching Realm that you would like to use. If no other Caching Realms are configured, you can safely choose the `defaultCachingRealm` entry.
 - e. Click the Apply button.
 - f. Restart the WebLogic Server.

Once the File Realm is configured and the WebLogic Server is restarted, the WebLogic Server will no longer use the Agent Realm. At this point, it is safe to uninstall the Agent using the supplied uninstaller. You may skip to the next section that details the steps necessary to use the supplied uninstaller. However, we recommend that you follow the remaining steps in this section to remove the Custom Realm and the Caching Realm that was created during the installation of the Agent.

3. **Agent Caching Realm removal:** After the File Realm has been configured not to use the Agent Caching Realm, the Agent Caching Realm can be safely removed from you system. This can be done by the following steps:
 - a. Logon to the WebLogic Server Administration Console. Use the configured system username and password for logging on to the console.
 - b. In the left pane of the WebLogic Server Administration Console, expand the Security Node by clicking on the “+” sign next to it.
 - c. In the left pane under Security node, click the Caching Realms. This forces the console to display a list of available Caching Realms in the system on the right pane.

- d. On the right pane, locate the row corresponding to the Agent Caching Realm. In the last column of this row, click on the delete icon image. A message is displayed to confirm the deletion of Agent Caching Realm. Read this message carefully to ensure that the Caching Realm that will be deleted will be Agent Caching Realm. Click on the yes button to confirm your decision. The console displays a confirmation message. Click on the Continue link located below this message.
- e. On the right pane, an updated list of Caching Realms will be displayed. Verify that there is no mention of Agent Caching Realm in this list.
- f. Restart the WebLogic Server.

Once the WebLogic Server has been restarted, you can remove the Agent Realm that was configured during the Agent installation.

4. **Agent Realm removal:** After the removal of Agent Caching Realm, the associated Custom Realm - Agent Realm can be removed. This can be done by the following steps:
 - a. Logon to the WebLogic Server Administration Console. Use the configured system username and password for logging on to the console.
 - b. In the left pane of the WebLogic Server Administration Console, expand the Security Node by clicking on the “+” sign next to it.
 - c. In the left pane under Security node, click the Realms. This forces the console to display a list of available Realms in the system on the right pane.
 - d. On the right pane, locate the row corresponding to the Agent Realm. In the last column of this row, click on the delete icon image. A message is displayed to confirm the deletion of Agent Realm. Read this message carefully to ensure that the Realm that will be deleted will be Agent Realm. Click on the yes button to confirm your decision. The console displays a confirmation message. Click on the Continue link located below this message.
 - e. On the right pane, an updated list of Realms will be displayed. Verify that there is no mention of Agent Realm in this list.
 - f. Restart the WebLogic Server.

Once the WebLogic Server is restarted, all the configuration settings that were made to the system during the installation of Agent have been removed. You can now proceed to the next section that describes how to use the Agent uninstaller to remove the Agent libraries from your system.

Launching the Uninstallation Program

Before launching the uninstaller program, make sure that the WebLogic Server is not running. Failing to do so can result in corruption of WebLogic startup which in turn can result in the system becoming unusable. When you are certain that the WebLogic Server is not running, you can launch the uninstallation program to remove Agent libraries and other files from your system.

The uninstallation program for Sun ONE Identity Server Policy Agent for WebLogic Server should be launched according to the following steps for Solaris, Windows, or HP-UX platform as applicable.

Launching the Uninstallation Program on Solaris 8

The uninstallation program for Solaris platform may be launched by executing the generated uninstall script located in the installation directory. The following steps provide details on how to uninstall the Agent:

1. Login as root.
2. Go to the directory where the Agent is installed.
3. Set your `JAVA_HOME` environment variable to JDK version 1.3.1 or higher. If your system does not have required version of JDK, use the JDK supplied with WebLogic 6.1 SP2 server. This JDK is located under:

WebLogic_Install_Dir/bea/jdk131

4. The uninstallation program provides two types of interfaces—a GUI and a command-line interface. In most cases, the GUI installer can be used for uninstalling the Agent. However, in cases when you are uninstalling the Agent over a telnet session on a remote server and do not have windowing capabilities, then it is recommended that you use the command line uninstallation program for uninstalling the Agent. You can launch this by executing the following command:

```
#./uninstall_wlagent -nodisplay
```

However, if you choose to use the GUI uninstallation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI uninstallation program window appears on the correct console.

NOTE If you choose to use the command line uninstallation program using the `-nodisplay` option, you may skip the next step and proceed directly to the next section, which details out the uninstallation procedure.

5. Launch the GUI uninstallation program by invoking the uninstall script as follows:

```
# ./uninstall_wlagent
```

The uninstallation program requires that you setup your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, if you have set the `JAVA_HOME` variable incorrectly, the uninstall script will prompt you for supplying the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the JDK installation directory that should be used for launching the installation program. Otherwise, enter a period (.) to abort the uninstallation.

In order that the GUI uninstallation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. In case your `DISPLAY` environment variable is not set at the time of invoking the uninstall script, the uninstallation program will prompt you for the `DISPLAY` environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

Provide the `DISPLAY` value to the installer by typing in the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

NOTE You can also use `uninstall_Sun_ONE_Identity_Server_Policy_Agent.class` file to uninstall the agent. You can find this file in the directory where you have installed the Agent.

Launching the Uninstallation Program on Windows 2000 Server

The uninstallation program for the Windows platform may be launched by executing the generated uninstall script located in the installation directory.

1. You must have administrative privileges when you run the uninstallation program. If you do not have administrative privileges, either login as “Administrator” or request such privileges to be granted to your account by the system administrator of the machine or domain as applicable.
2. Go to the directory where Agent is installed.
3. The uninstall script `uninstall_wlagent.bat` is located in this directory. In order to use the `uninstall_wlagent.bat` script to launch the uninstallation program, you must have JDK version 1.3.1 or higher. This can be verified by typing the following command in the command prompt window:

```
C:\> java -version

java version "1.3.1_02"

Java(TM) 2 Runtime Environment, Standard Edition (build
1.3.1_02-b02)

Java HotSpot(TM) Client VM (build 1.3.1_02-b02, mixed mode)
```

If you do not have JDK of required version in your system path, you can use the JDK supplied with WebLogic 6.1 SP2 server located at:

WebLogic_Install_Dir\bea\jdk131

Once this is done, the `uninstall_wlagent.bat` may be executed by typing the file name at the command prompt window in a directory where it is present, or by double clicking the file in Windows Explorer. For example:

```
C:\Sun>uninstall_wlagent.bat
```

The uninstallation program provides two types of interfaces—a GUI and a command line interface. By invoking the `uninstall_wlagent.bat` file from a command prompt window as shown above or by double clicking it in Windows Explorer, the uninstallation program is launched in the GUI mode. However, in a case where it is required that you use the command line uninstallation program for uninstalling the Agent, the uninstallation program may be launched by executing the `uninstall_wlagent.bat` file and passing in a command line argument `-nodisplay` as follows:

```
C:\Sun>uninstall_wlagent.bat -nodisplay
```

This can be done by typing the above command in a command prompt window in a directory where this file is available.

NOTE

- If the required JDK is configured to be in the system path, the uninstallation program can be launched from the Control Panel Add/Remove programs control. In the list of the installed programs on your system, select Sun ONE Identity Server Policy Agent for WebLogic and click the Change/Remove button. If your system path is not configured with an appropriate JDK version, the uninstall can fail.
 - When using the Control Panel Add/Remove programs control to launch the uninstallation program for Agent, the uninstallation program will be launched in GUI mode only. To launch the uninstallation program in command line mode, use the provided uninstall script as mentioned previously.
-

Launching the Uninstallation Program on HP-UX 11

The uninstallation program for HP-UX platform may be launched by executing the generated uninstall script located in the installation directory. The following steps provide details on how to achieve this:

1. Login as root.
2. Go to the directory where the Agent is installed.
3. Set your `JAVA_HOME` environment variable to a JDK version 1.3.1 or higher. If your system does not have JDK of required version, use the JDK supplied with WebLogic 6.1 SP2 server. This JDK is located under:
WebLogic_Install_Dir/boa/jdk131
4. The uninstallation program provides two types of interfaces— a GUI and a command-line interface. In most cases, the GUI installer can be used for uninstalling the Agent. However, in cases when you are uninstalling the Agent over a telnet session on a remote server and do not have windowing

capabilities, it is recommended that you use the command line uninstallation program for uninstalling the Agent. This can be launched by executing the `uninstall` script and passing in a command line argument `-nodisplay` as follows:

```
#./uninstall_wlagent -nodisplay
```

However, if you choose to use the GUI uninstallation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI uninstallation program window appears on the correct console.

NOTE If you choose to use the command line based uninstallation program using the `-nodisplay` option, you may skip the next step and proceed directly to the next section, which details out the uninstallation procedure.

5. Launch the GUI uninstallation program by invoking the `uninstall` script as follows:

```
# ./uninstall_wlagent
```

The uninstallation program requires that you setup your `JAVA_HOME` variable correctly as pointed out in the step number 3. However, in case you have incorrectly set the `JAVA_HOME` variable, the `uninstall` script will prompt you for supplying the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the JDK installation directory that should be used for launching the installation program. Otherwise, enter a period (.) to abort the uninstallation.

In order that the GUI uninstallation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. In case your `DISPLAY` environment variable is not set at the time of invoking the `uninstall` script, the uninstallation program will prompt you for the `DISPLAY` environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

Provide the `DISPLAY` value to the installer by typing in the exact value at the above prompt. Otherwise, enter a period (.) to abort the uninstallation.

Uninstalling the Agent Using GUI

The uninstallation program begins with a Welcome screen. Click Next to step through the uninstallation screens, answering the questions.

1. In the Uninstall Type Selection Screen select Full and click Next.
2. In the Ready to Uninstall screen, review the uninstallation information. If you need to make changes, click Back. Otherwise, click Uninstall Now.
3. The Uninstall Progress screen displays the progress of uninstall process.
4. In the Uninstallation Summary window, click Details for a detailed summary of the configuration information that was processed during uninstallation. Click Exit to end the program

NOTE If the status of the uninstall is “Failed”, you must view the log file for the uninstall by clicking on the Details button to identify which uninstall task failed. In certain situations these failures can be recovered from and the system can be restored to its original state. Refer to the next section for the detail of how to restore the system to its original state.

Troubleshooting Uninstallation Problems

During the installation, the Agent Installer performs modifications for certain existing files on your system. In order that these modifications can be backed out completely, backups of these files are created before the modifications are made. In case you encounter any failures during the uninstall, it is possible to bring the system back to its original state manually by restoring the backed up files.

The following is a list of files that are backed up during installation and restored during the uninstall of the Agent:

- WebLogic Server Startup Script
- java.security file

WebLogic Server Startup Script

The WebLogic Startup script is modified during installation and the backup is created in the same directory as the startup script. The name of the backup file is *WebLogic_Startup_Script_Name-preAgent*. For example, if the WebLogic startup script that you specified during installation was:

```
/bea/wlserver6.1/config/examples/startExamplesServer.sh
```

then the backup file will be:

```
/bea/wlserver6.1/config/examples/startExamplesServer.sh-preAgent
```

java.security file

The Agent installer installs the JCE and JSSE extensions to the JDK installation being used by WebLogic Server. These extensions require that the specific providers be entered into the JDK installations security file so that they are available at runtime. Before these providers are entered into the security file, a backup of the security file is created so that it can be restored during the uninstall of the Agent. The backup file is named `java.security-preAgent`, and is located in the JDK installation's `jre/lib/security` directory. For example, if during the install you entered the WebLogic `JAVA_HOME` value as `/bea/jdk131`, the `java.security` file will be:

```
/bea/jdk131/jre/lib/security/java.security
```

and the backup file will be:

```
/bea/jdk131/jre/lib/security/java.security-preAgent
```

Policy Agent 2.0 for IBM WebSphere 4.0.4 AE

This document provides a brief overview of Sun ONE Identity Server Policy Agent for IBM WebSphere Application Server 4.0.4 and describes how to install and configure the Policy Agent for WebSphere 4.0.4 running on Solaris 8 platform.

Topics include:

- Overview
- Guidelines
- Limitations
- Software Requirements
- Installing the Agent
- Configuring WebSphere Application Server
- Uninstalling the Agent
- Troubleshooting Information

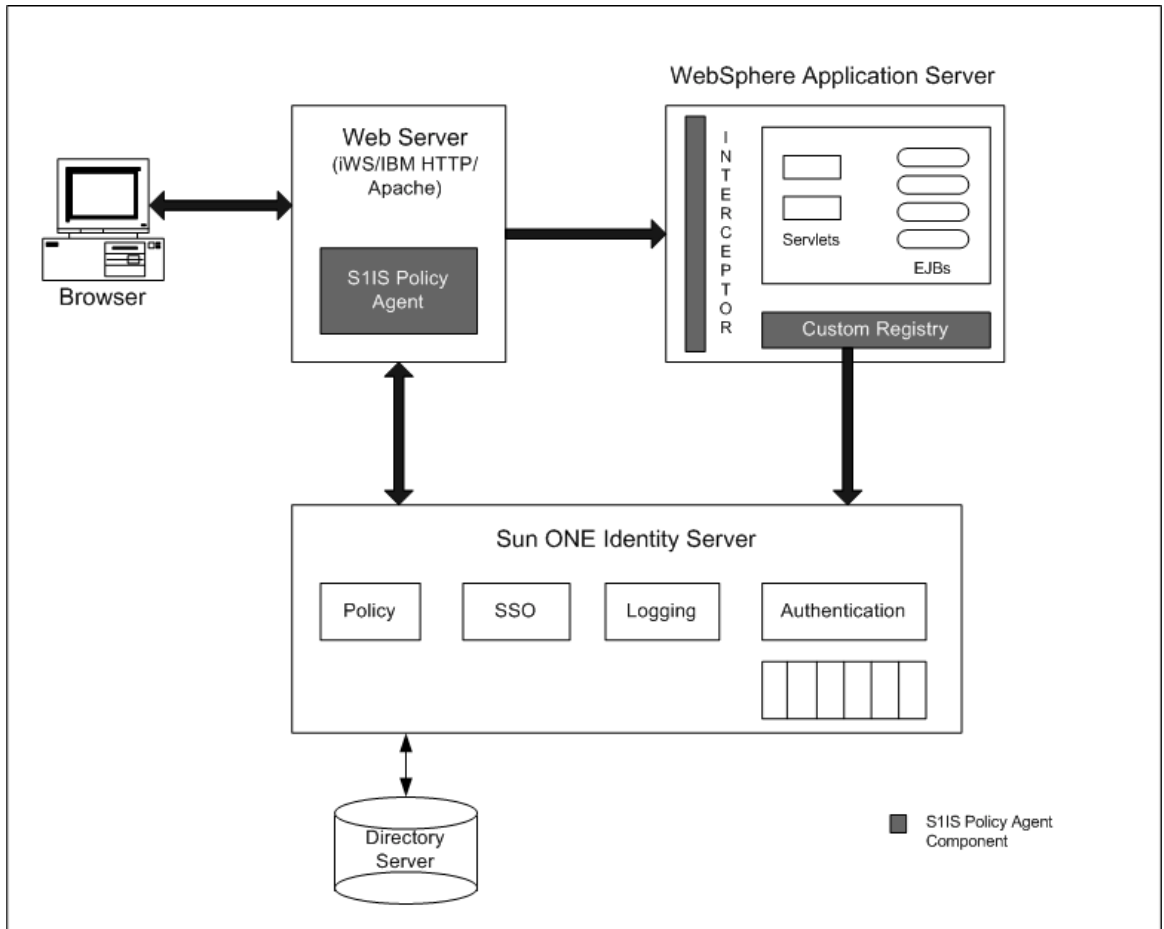
Overview

The Identity Server Policy Agent for WebSphere Application Server 4.0.4 enforces authentication and authorization for clients that interact with the application through the application server's web container. The resources deployed on the WebSphere are protected by a combination of a Web Agent and a J2EE Agent.

The Identity Server Web Agent is used as a Reverse Proxy Security Server (RPSS) to perform the authentication. The RPSS authenticates the HTTP clients and passes authenticated requests to the WebSphere Application Server. The Identity Server Policy Agent is installed on the Web server (IBM HTTP, iWS, or Apache) that authenticates clients for the WebSphere Application Server. The Web Agent is used to perform authentication by redirecting the request for protected resources to the Identity Server Authentication service. The Web Agent also performs coarse grained authorization based on the Allow or Deny URL Policies defined in the Identity Server.

The J2EE authorization model is based on the concept of security roles. A security role is a logical grouping of users defined by an Application Component Provider or Assembler. Application deployer maps roles to security identities (for example, principals and groups) in the user registry. Security roles are used with declarative security (through deployment descriptor) and programmatic security. The WebSphere policy Agent enables the mapping of Identity Server security identities such as Identity Server users and Identity Server roles, to the security roles used within the J2EE Application using the Custom User Registry implementation for WebSphere 4.0.4.

The J2EE role-based authorization is performed by the WebSphere application server. The WebSphere server must trust the RPSS before accepting the authentication credentials. For WebSphere to trust the RPSS, a Web Trust Association (WTA) between WebSphere and RPSS needs to be established. This can be achieved by implementing the Web Trust Association Interceptor (WTAI) for the RPSS. The WTAI implementation establishes the trust by validating SSO Token in the request. On a successful trust validation, the WebSphere runtime uses the WTAI to retrieve principal from the request. The WebSphere runtime uses the identity information obtained from the WTAI to communicate with the user registry. The user registry is the custom registry that communicates with the Identity Server. WebSphere runtime invokes the methods of the user registry to get the set of security identities applicable for the current user. The security identities are used to determine the security roles applicable for the user. This information is used by WebSphere to implement both declarative and programmatic security.

Figure 8-1 Policy Agent for WebSphere 4.0.4 AE Architecture

Guidelines

The following guidelines will help you use the Identity Server Policy Agent for WebSphere most optimally:

Agent-based Authentication and Authorization

After the agent is installed and the application has been secured, the web agent enforces authentication for all web based access to the protected application's enforced portions. Working in tandem with the Agent Realm (Custom Registry) component, the Agent Interceptor ensures that the J2EE policies defined for the protected application get evaluated correctly based on the set role-to-principal mappings, at the same time offering other key services such as Single Sign-On. Therefore, it is recommended that protected applications do not use their own authentication mechanism or any container-based authentication mechanisms such as Basic and Form Authentication.

Coarse Grained and Fine Grained Access-control

The resources deployed on WebSphere are protected by a combination of a Web Agent and a J2EE Agent. The Web Agent is primarily used to perform authentication by redirecting the request for protected resources to the Identity Server Authentication service. It also performs coarse grained authorization based on the Allow or Deny URL Policies defined in Identity Server.

The J2EE Agent assists the WebSphere Application server to perform more fine grained access-control for individual servlets, EJBs, or EJB methods based on the security constraints and EJB method permissions defined in the deployment descriptors. The J2EE Agent maps the application-defined security roles to the principals defined in Identity Server.

The resources deployed on WebSphere Application server can be protected using either coarse grained, or fine grained access control, or a combination of both. For example, an employee portal wants to provide read access to any authenticated (regular) employee to its payroll web application, and modification or write access to only certain people. The first requirement can be achieved by defining a Allow URL Policy in the Identity Server for the application URL, applicable for the entire organization. This is coarse grained access-control. The second requirement can be met by defining the security constraint for the servlet or EJB method that performs the update or modification. The security role associated with the constraint is then mapped to privileged users and roles in the Identity Server. Access to the EJBs or servlets is then restricted only to these users. This is fine grained access-control.

The roles in the security constraint (mentioned as J2EE roles) can be mapped to either Identity Server users, or Identity Server roles. This helps achieve role-user mapping dynamically or at runtime. A user added to the Role in the Identity Server will get access to the resources protected by the J2EE role, which is mapped to this Identity Server role, immediately without having to restart the WebSphere Application Server. Similarly once the user is removed from the role, the user is automatically denied the access to those resources; the Application or WAS need not be restarted.

However, the mapping of the J2EE roles to the Identity Server principals (Users and Roles) itself is static. This mapping is initially specified during the Application Deployment. However, the mapping can also be changed for the already deployed application through the WebSphere 4.0 Administrative Console.

But for these changes to take effect, the application (individual application only, and not the Application Server) needs to be restarted from the WebSphere Administrative Console.

Create Enhanced Security Aware Applications

The Agent provides the rich APIs offered by Identity Server SDK libraries, which are available for use within the protected application. Using these APIs, the application architect can create enhanced security aware applications that are customized to work in the security framework offered by Identity Server. For more information on how to use the Identity Server SDK, refer to the Sun ONE Identity Server Programmer's Guide.

Limitations

- The Policy Agent for WebSphere Application Server 4.0.4 is designed to enforce authentication and authorization for clients that interact with the application through the application server's web container. Typically, such clients are thin clients like web browsers that communicate with the application server's web container using HTTP or HTTPS protocols. If however, the client does not access the application through the HTTP or HTTPS protocols, thereby bypassing the web container of the application server, the Agent Interceptor component will not be in a position to intercept the request. This is a likely case when the application is being accessed by a rich client using other protocols such as RMI over IIOP. Since the Agent Interceptor component is bypassed in such cases, the authentication cannot be enforced. Clients that do not get authenticated by the Agent Interceptor do not

possess sufficient credentials for the Agent Realm component to evaluate the necessary J2EE security policies. Therefore, such clients will be denied access to all protected resources. Alternatively, for security aware applications that use the programmatic J2EE security APIs provided in the application server will be given negative results by such APIs.

- When trust association is configured between a WebSphere Application Server and a Reverse Proxy Security Server, access to all protected resources will go through trust association. The Trust Association is bypassed for access to unprotected resources. The WebSphere Application Server does not invoke the Trust Association Interceptor when access is made to unprotected resources. If an unprotected resource accesses a protected resource, the access will always fail, even if the resource was accessed from an authenticated session, which has the roles for accessing the protected resource.

Workaround

All unprotected resources, which access protected resources must be protected with a dummy security constraint. Access to these resources should be limited to any Authenticated User, by mapping the role associated with the dummy security constraint to the special subject "All Authenticated Users." This will ensure that the Trust association is not bypassed when these resources are accessed. However, with this workaround the unprotected resource no longer remains truly unprotected.

Supported Platform

Solaris 8

Software Requirements

- Sun ONE Identity Server 6.0
- One of the following Web Servers with an appropriate Identity Server Policy Agent 2.0 installed
 - Apache Server 1.3.26
 - IBM HTTP Server 1.3.19
 - iPlanet Web Server, Enterprise Edition 4.1 SP8
- WebSphere Application Server 4.0.4 Advanced Edition

Installing the Agent

The Identity Server Policy Agent for WebSphere 4.0.4 AE is supported on Solaris 8 platform.

Pre-Installation Tasks

The following tasks must be completed before you install the Identity Server Policy Agent for WebSphere:

1. Install the database for the WebSphere Application Server. For information on installing the database, see the documentation at:

http://www-3.ibm.com/software/webservers/appserv/doc/v40/ae/info-center/was/nav_solaris.html

2. Install one of the following Web Servers:

- iPlanet Web Server 4.1 SP8
- Apache 1.3.26
- IBM HTTP Server 1.3.19 (bundled with WebSphere)

3. Install WebSphere Application Server. See installation details at:

http://www-3.ibm.com/software/webservers/appserv/doc/v40/ae/info-center/was/nav_solaris.html

4. Install Fixpacks for WebSphere and IBM HTTP Server.

Version upgrade to 4.0.4

1. Install Fixpack4 for WebSphere Application Server, to upgrade the version from 4.0.1 to 4.0.4. Download Fixpack4 from:

<ftp://ftp.software.ibm.com/software/websphere/appserv/support/fixpacks/was40/fixpack4/Sun/>

2. When you apply the Fixpack, choose to apply the changes to all the components (IBM HTTP Server, bundled JDK, and WebSphere Application Server.)

3. Test your installation and configuration of WebSphere Application Server. Follow the instructions at:

<http://www-3.ibm.com/software/webservers/appserv/doc/v40/ae/info-center/was/022soltestgen.html>

4. Install the appropriate Identity Server Policy Agent for the Web Server.

If only fine grained authorization is performed based on the deployment descriptors, then install the Identity Server Policy agent with the Single Sign On (SSO) Only option. With this option, the Web Server Policy Agent performs SSO only, and doesn't perform authorization based on URL policies defined in the Identity Server. For more details, refer the chapter "Read This First," on page 19.

5. Add Unprotected resources to the Not-Enforced List of the Web Agent.

The Web Agent intercepts the request to the web server. It checks for the presence of valid a SSO Token in the request. If a valid SSOToken is not found in the request, the user is redirected to the Identity Server for authentication. Requests to the resources deployed on WebSphere go through the Web Server and are intercepted by the Web Agent. Some applications deployed on the WebSphere may not be protected and may be accessed by unauthenticated users. For such applications, the entries need to be created in the Not-Enforced List property of the Web agent configuration file. With this configuration, the web agent no longer forces authentication to the unprotected resources specified and access to these resources is allowed unconditionally. See the chapter "Read This First," on page 19 for more details.

6. Enable Cookie Reset Feature in the Web Agent

The LTPA authentication framework in WebSphere sets LtpaToken cookie in the browser session on successful authentication. This identifies the authenticated user to the WebSphere Application Server for the requests in that session. When Web Agent is installed, the authentication is managed by Identity Server. When a user session ends in Identity Server, the LtpaToken cookie in the request no longer identifies the authenticated user. Due to the continued presence of LtpaToken, WebSphere continues the execution in the context of the authenticated user represented by the LtpaToken. To clear the information in the LtpaToken cookie, enable the cookie reset feature. This can be done from the web agent by setting the following configuration parameters in the Web Agent's configuration file `AMAgent.properties` as shown below.

Enable Cookie Reset if it is not already enabled.

```
com.sun.am.policy.agents.cookie_reset_enabled=true
```

Append LtpaToken to the list of cookies to be reset

```
com.sun.am.policy.agents.cookie_reset_list=LtpaToken
```

7. Create URL Policies for the resources in Identity Server.

This step is required only if you choose to perform coarse grained access-control based on the Allow/Deny URL policies in the Identity Server. If you have chosen to use the Sun ONE Identity Server Policy agent to perform only the single sign on in step 4 then this step can be skipped.

Create Allow Policies for resources, for which you don't want to perform any coarse grained access control, but only fine grained access control through security constraints and EJB method permissions in deployment descriptor. You can also create the Deny policies to Deny certain resources to subjects based on conditions. By default if no policy is created, access to all resources is blocked by the Web Agent.

For example, an Allow policy for URL

`http://webserverhost.domain:port/*` assigned to subject organization will cause the Web Agent to allow all the requests to pass through.

8. Test the deployment of the application that you intend to protect.

Before installing the WebSphere Agent, it is important that you deploy and test the application that must be protected for simple functionality. Once it is established that the application can be deployed successfully, you are ready to install the Agent.

9. Disable Security for the WebSphere Application Server, if already enabled.

NOTE

Security needs to be disabled only during the agent installation, The agent installation program re-enables the security after installing the custom user registry. This step is required for the agent installation program to install and configure the custom User Registry.

This can be done from the WebSphere Administrative Console as follows.

10. Start the WebSphere Application Server and then start the WebSphere Administrative Console.

```
# WAS_root_dir/bin/adminclient.sh
```

- Choose Console > Security Center.
- In the Security Center Window, click on General tab.
- Ensure that the check box Enable Security is not selected.
- Click Apply.

- Click OK.

11. Restart the WebSphere Administration Server.

NOTE The Agent installation program updates the Security Configuration, and installs the Agent Realm as the User Registry. If the WebSphere is configured to use the OS realm, LDAP realm, or any other custom registry, the Agent installation program will not be able to authenticate to the current realm, subsequently the Agent Realm installation and Security Configuration will fail during the agent installation.

Launching the Installation Program

The binaries for Identity Server Policy Agent for WebSphere 4.0.4 AE for Solaris platform are provided as a tar-gzip archive. Copy this archive to the machine where WebSphere Application Server is installed. Perform the following steps to launch the installation program.

1. Login as `root`.
2. Unzip the binary archive using the following command:

```
# /bin/gzcat IBM_WebSphere_4.0.4_agent_2.0_Solaris2.8.tar.gz |
/bin/tar xvf -
```

3. Set your `JAVA_HOME` environment variable to the JDK supplied with WebSphere Server 4.0.4 server. This JDK is typically located at:

```
WAS_root_dir/java
```

The installation program provides two types of interfaces: a graphical user interface (GUI) and a command-line interface. In most cases, the GUI installation program can be used for installing the Agent. However, in cases when you are installing the Agent over a telnet session on a remote server, and do not have windowing capabilities, it is recommended that you use the command-line installation program for installing the Agent. You can launch this by executing the following command:

```
# ./setup -nodisplay
```

However, if you choose to use the GUI installation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI installation program window appears on the correct console.

4. Launch the GUI installation program by invoking the setup script as follows:

```
# ./setup
```

The installation program requires that you set up your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, if you have set the `JAVA_HOME` variable incorrectly, the setup script will prompt you for the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

5. Type the full path to the WebSphere JDK installation directory for launching the installation program. This typically is `WAS_root_dir/java`. Otherwise, enter a period (.) to abort the installation.

In order that the GUI installation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly.

If your `DISPLAY` environment variable is not set at the time of invoking the setup script, the installation program will prompt you for the `DISPLAY` environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

6. Provide the `DISPLAY` value to the installation program by typing the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

Installing the Agent Using GUI

The installation program begins with a Welcome screen.

1. Click Next to step through the installation screens, and answer the required questions.
2. Read the License Agreement. Click Yes (Accept License) to continue with the Installation.
3. In the Select Installation Directory screen, enter the path where you want to install. The default installation directory is `/opt`.

Figure 8-2 Select Installation Directory screen



4. Click Next and in the Sun ONE Identity Server Information screen, provide the following information about the Identity Server.

Figure 8-3 Sun ONE Identity Server Information Screen

Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wiz

Sun(TM) ONE Identity Server Information

Enter the server information where the Sun(TM) ONE Identity Server Service is installed.

Sun(TM) ONE Identity Server Host: nila.Eng.Siroe.COM

Sun(TM) ONE Identity Server Port: 58080

Sun(TM) ONE Identity Server Protocol: ☒ http ☐ https

Sun(TM) ONE Identity Server Deployment URI: amserver

amAdmin Password: ****

Re-enter Password: ****

amldapuser Password: ****

Re-enter Password: ****

Back Next Cancel Help

Sun ONE Identity Server Host: Enter the fully qualified host name of the system where Sun ONE Identity Server is installed. For example, nila.eng.siroe.com.

Sun ONE Identity Server Port: Enter the port number for the Web Server that runs Sun ONE Identity Server Services. The default port number is 58080.

Sun ONE Identity Server Protocol: Select the protocol that will be used by the Agent to communicate with Identity Server services. If the web server has been configured for SSL, select HTTPS; otherwise, select HTTP.

Sun ONE Identity Server Deployment URI: Enter the URI that should be used for accessing Identity Server Services. The default URI is amserver.

amAdmin Password: Enter the password for the amAdmin user.

Re-enter Password: Re-enter the password for the amAdmin user for confirmation.

amldapuser Password: Enter the password for the amldapuser. This is the Bind DN user for LDAP/Membership/Policy service. This user will have read and search access to Directory Server entries.

Re-enter Password: Re-enter the password for the amldapuser for confirmation.

NOTE The amAdmin password supplied during installation is recorded by the Agent in a secure manner. However, if in the future you change this password in Identity Server, you will have to update the password in the Agent as well. This can be done by using the `agentadmin` tool provided with the Agent. Once the Agent has been installed on the system, the `agentadmin` tool can be invoked from the location:

`Agent_Install_Dir/SUNWam/wasAgent/bin/agentadmin`

To change the password, invoke this tool as follows:

`#./agentadmin -password oldpassword newpassword`

Restart WebSphere for changes to take effect.

5. Click Next and in the Directory Server Information screen, provide the following information about the Directory Server that is associated with Identity Server services.

Figure 8-4 Directory Information Screen

Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install

Directory Information

Enter the directory information corresponding to the Sun(TM) ONE Identity Server Services.

Directory Host: nila.Eng.Siroe.COM

Directory Port: 389

SSL Enabled : ☐

Root Suffix: dc=siroe,dc=com

Installation Organization: dc=siroe,dc=com

Back Next Cancel Help

Directory Host: Enter the fully qualified host name of the system where the Directory Server is installed. For example, nila.eng.siroe.com.

Directory Port: Enter the port number used by the Directory Server. The default port is 389.

SSL Enabled: Select this, if the Directory Server instance used by the Identity Server is configured for LDAPS.

Root Suffix: Enter the root suffix to be used with this Directory Server.

Installation Organization: Enter the name of the installation organization as specified when installing the Sun ONE Identity Server.

6. In the WebSphere Application Server Details screen, provide the following information about the WebSphere Server on which the Agent is installed.

Figure 8-5 WebSphere Application Server Details Screen

The screenshot shows a Windows-style installation wizard window titled "Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard". On the left is a vertical sidebar with the Sun Microsystems logo and the text "Sun™ ONE Identity Server". The main area is titled "WebSphere Application Server Details" and contains the instruction "Enter WebSphere Application Server details." Below this are three input fields: "WebSphere Application Server Root Directory." with the value "/opt/WebSphere/AppServer", "WebSphere Application Server JAVA_HOME directory" with the value "/opt/WebSphere/AppServer", and "Name of the Agent Realm." with the value "AgentRealm". At the bottom are four buttons: "Back", "Next", "Cancel", and "Help".

WebSphere Application Server Root Directory: Enter the full path to the location of the WebSphere root directory. For example, *agent_install_dir/WebSphere/AppServer*

WebSphere Application Server JAVA_HOME Directory: Enter the full path to the JDK installation directory used by the WebSphere Server. The WebSphere JAVA_HOME directory refers to the JDK installation that is used by the WebSphere Server. By default, this is *WAS_root_dir/java*

Name of the Agent Realm: Enter the name of the Realm that will be configured as the default realm to be used by the WebSphere Application Server.

- Click Next and in the Agent Configuration Details screen, provide the key configuration information necessary for the Agent to function correctly.

Figure 8-6 Agent Configuration Details Screen

Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard

Agent Configuration Details

Enter Agent configuration values.

Agent Audit Log File: /var/opt/SUNWam/audit/agent.log

Enable Audit log file Rotation: ☐

UnAuthenticated User: anonymous

Enable Console Integration: ☐

Enable Not-Enforced List Cache: ☐

Number of Entries in Cache: 1000

Cache Expiration Time in Seconds: 180

Back Next Cancel Help

Agent Audit Log File: Enter the complete path to the log file to be used by the agent to record Audit messages. The agent logs key events such as user authentication in this log file.

Enable Audit log file rotation: Select this to enable rotation of Audit Log files. Selecting this ensures that after the log file reaches a specified size, the agent creates a new file for logging audit messages. The audit log file rotation size is controlled by the property Audit Log File Rotation Size in the `AMAgent.properties` file.

UnAuthenticated User: Enter the User id of the unauthenticated user. This user will be used when protected resources are included in the Not-Enforced List. The Agent Interceptor returns this as the user to be used for accessing the protected resources included in the Not Enforced List.

Enable Console Integration: Select this option to enable console level integration of Sun ONE Identity Server with WebSphere Administrative Console.

Enable Not-Enforced List Cache: Select this option to enable caching of Not-Enforced List evaluation results.

Number of Entries in Cache: Specify the number of entries that cache can hold at a given instance.

Cache Expiration Time: Specify the time in seconds to be used as the limit for entries that can exist in the Not-Enforced List cache.

Agent Configuration Details

The Audit Log file is a necessary requirement for the Agent. You may provide the name of a non-existing file on the system to be used as the Audit file. In which case, the Agent creates this file and the necessary directories in its path on first use. Alternatively, you can provide the filename of an existing file which can be used by the Agent. However, in either case, it is required that the specified file has write permissions for the WebSphere Server process because the Agent executes in the same process as the WebSphere Server. Providing an incorrect value for this will result in the malfunction of the Agent which can render the WebSphere Server unusable.

Console Integration implies that the information regarding the names of all Users, Roles, and the Users who are assigned to these Roles in Identity Server will be visible on the WebSphere Server Administrative Console. Therefore, this feature must be enabled with caution since the User or Role information will then be accessible to administrators who access the WebSphere Server Console.

The Agent Interceptor is bypassed for access to unprotected resources. The unprotected resources are the resources which are not protected by security constraints in the deployment descriptor. Since the Not-Enforced List is processed by the Agent Interceptor, this will not impact the unprotected resources. Therefore, it is not necessary to add any unprotected resources to the Not-Enforced List. The resources in the Not-Enforced List will always be accessed as the unauthenticated user—for example, anonymous, independent of the identity of the logged in user.

The Application Server authorization enforcement takes precedence over the Not-Enforced List configuration specified in `AMAgentConfiguration.properties`. The access to the protected resource

specified in Not-Enforced List will be successful, only if the unauthenticated user, has the appropriate roles to access the application based on the application's deployment descriptors.

When the Agent must process a large list of Not-Enforced pattern rules specified in the configuration, every incoming request must be evaluated against every such rule to determine if the request can be allowed without authentication. In scenarios where the user load is high, the time spent in evaluating these rules can add up and degrade the overall performance of the system. To avoid this problem, it is recommended that Not-Enforced List Cache be enabled.

Enabling the Not-Enforced List Cache may result in degraded performance if the values for Number of Entries in the Cache and Cache Expiration time are not set up appropriately. In cases when the cache expiration time duration is more than necessary, the cache will get filled up very fast and new requests will still be evaluated against all the specified pattern rules, resulting in no improvement in system performance. If the number of entries in Cache is set very high, it may result in excessive consumption of system memory, thereby leading to degraded performance. Therefore, these values must be set up after careful deliberation of the deployment scenario and should be changed as necessary to reflect the changing usage scenario of the system. It is recommended that the system be tested with various values of these two parameters in a controlled environment to identify the optimal values, and only then be deployed in production.

The Agent maintains two caches in its memory, one for recording the URIs that were evaluated as enforced, and the other for recording the URIs that were evaluated as not-enforced. The specified values of Number of Entries in Cache and Cache Expiration time is equally applicable to both of these caches. This factor must be considered when setting the values for the size and expiration time of the cache.

8. In the Summary of all the selections screen, review and verify the installation options that you have specified for the Agent. If you need to make changes, click Back. Otherwise, click Next to proceed.
9. In the Ready to Install screen, click Install Now to begin the installation.

The Install Progress screen displays the progress of the installation as the installation program makes changes to your system. If necessary, you may interrupt this process by clicking the Stop button.

NOTE It is strongly recommended that you do not interrupt this process, as this may lead to a partially installed product. This may also cause problems with uninstall, and may render the WebSphere Server unusable. This process should be interrupted only in cases where it is absolutely necessary to do so.

10. In the Installation Summary screen, click on Details for a detailed summary of the configuration information that was processed during installation. Click on Exit to end the program.

NOTE It is recommended that you check the install log file by clicking the Details button to identify errors. If you find errors, uninstall the Agent and try again after fixing the root cause of failure during this installation.

11. Start WebSphere Application Server using the following command:

```
# WAS_root_dir/bin/startupServer.sh
```

After the agent is installed, the next step is to configure the WebSphere Server as explained in the later sections.

Installing the Agent Using Command-Line

You must have root permissions when you run the agent installation program.

1. Unzip the Solaris tar file using the following command:

```
# gzcat IBM_WebSphere_4.0.4_agent_2.0_Solaris2.8.tar.gz | tar  
-xvf -
```

2. Run the `setup` program. You'll find the program in the directory where you untarred the binaries. At the command-line, enter the following:

```
# ./setup -nodisplay
```

3. When prompted to enter path to pick up java, type the full path where the WebSphere JDK is located. This is typically `WAS_root_dir/java`

4. When prompted provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement? Type Yes.

5. At the following prompt, specify an installation directory for the agent.

The Sun(TM) ONE Identity Server Policy Agent components will be installed in the following directory, which is referred to as the "Installation Directory". To use this directory, press only the Enter key. To use a different directory, type in the full path of the directory to use followed by pressing the Enter key.

Install Sun(TM) ONE Identity Server Policy Agent in this directory [/opt] {"<" goes back, "!" exits}:

Install Sun(TM) ONE Identity Server Agent in this directory: Enter the full path to the directory in which you want to install the Policy Agent.

6. At the following prompts, provide details about your Identity Server installation.

```
Sun(TM) ONE Identity Server Services Details.
  Sun(TM) ONE Identity Server Host [nila.Eng.Siroe.COM] {"<" goes back, "!"
exits}:
  Sun(TM) ONE Identity Server Port [58080] {"<" goes back, "!" exits}:
  Sun(TM) ONE Identity Server Protocol [http] {"<" goes back, "!" exits}:
  Sun(TM) ONE Identity Server Deployment URI [/amserver] {"<" goes back,
"!" exits}:
  amAdmin Password [] {"<" goes back, "!" exits}:
  Re-enter Password [] {"<" goes back, "!" exits}:
  amldapuser Password [] {"<" goes back, "!" exits}:
  Re-enter Password [] {"<" goes back, "!" exits}:
```

Sun ONE Identity Server Host: Enter the fully qualified host name of the system where Sun ONE Identity Server is installed. For example, nila.eng.siroe.com.

Sun ONE Identity Server Port: Enter the port number for the Web Server that runs Sun ONE Identity Server Services. The default port number is 58080.

Sun ONE Identity Server Protocol: Select the protocol that will be used by the Agent to communicate with Identity Server services. If the web server has been configured for SSL, select HTTPS; otherwise, select HTTP.

Sun ONE Identity Server Deployment URI: Enter the URI that should be used for accessing Identity Server Services. The default URI is `amserver`.

amAdmin Password: Enter the password for the amAdmin user.

Re-enter Password: Re-enter the password for the amAdmin user for confirmation.

amldapuser Password: Enter the password for the amldapuser. This is the Bind DN user for LDAP/Membership/Policy service. This user will have read and search access to Directory Server entries.

Re-enter Password: Re-enter the password for the amldapuser for confirmation.

NOTE The amAdmin password supplied during installation is recorded by the Agent in a secure manner. However, if in the future you change this password in Identity Server, you will have to update the password in the Agent as well. This can be done by using the `agentadmin` tool provided with the Agent. Once the Agent has been installed on the system, the `agentadmin` tool can be invoked from the location:

`Agent_Install_Dir/SUNWam/wasAgent/bin/agentadmin`

To change the password, invoke this tool as follows:

`#./agentadmin -password oldpassword newpassword`

Restart WebSphere for changes to take effect.

7. At the following prompts, provide information about the Directory Server used by the Identity Server.

```

Sun(TM) ONE Identity Server Directory Details
Directory Host [nila.eng.Siroe.COM] {"<" goes back, "!" exits}:
Directory Port [389] {"<" goes back, "!" exits}:
SSL Enabled [false] {"<" goes back, "!" exits}:
Root Suffix [dc=siroe,dc=com] {"<" goes back, "!" exits}:
Installation Organization [dc=siroe,dc=com] {"<" goes back, "!"
exits}:

```

Directory Host: Enter the fully qualified host name of the system where the Directory Server is installed. For example, nila.eng.siroe.com.

Directory Port: Enter the port number used by the Directory Server. The default port is 389.

SSL Enabled: Select this, if the Directory Server instance used by the Identity Server is configured for LDAPS.

Root Suffix: Enter the root suffix to be used with this Directory Server.

Installation Organization: Enter the name of the installation organization as specified when installing the Sun ONE Identity Server.

8. At the following prompts, enter the details of WebSphere Application Server.

Enter the details of WebSphere Application Server installation on which this Agent will be installed.

```

WebSphere Application Server Root Directory.
[/opt/WebSphere/AppServer] {"<" goes back, "!" exits}
WebSphere Application Server JAVA_HOME directory
[/opt/WebSphere/AppServer/java] {"<" goes back, "!" exits}
Name of the Agent Realm. [AgentRealm] {"<" goes back, "!" exits}

```

WebSphere Application Server Root Directory: Enter the full path to the location of the WebSphere root directory. For example,
agent_install_dir/WebSphere/AppServer

WebSphere Application Server JAVA_HOME Directory: Enter the full path to the JDK installation directory used by the WebSphere Server. The WebSphere JAVA_HOME directory refers to the JDK installation that is used by the WebSphere Server. By default, this is *WAS_root_dir/java*

Name of the Agent Realm: Enter the name of the Realm that will be configured as the default realm to be used by the WebSphere Application Server.

9. Enter the configuration details for the installation of Sun ONE Identity Server Agent for WebSphere Application Server.

```
Agent Audit Log File [/var/opt/SUNWam/audit/agent.log] {"<" goes back, "!" exits}
Enable Audit log file Rotation [false] {"<" goes back, "!" exits}
Enable Console Integration [false] {"<" goes back, "!" exits}
Enable Not-Enforced List Cache [false] {"<" goes back, "!" exits}
UnAuthenticated User [anonymous] {"<" goes back, "!" exits}
```

Agent Audit Log File: Enter the complete path to the log file to be used by the agent to record Audit messages. The agent logs key events such as user authentication in this log file.

Enable Audit log file rotation: Select this to enable rotation of Audit Log files. Selecting this ensures that after the log file reaches a specified size, the agent creates a new file for logging audit messages. The audit log file rotation size is controlled by the property Audit Log File Rotation Size in the *AMAgent.properties* file.

UnAuthenticated User: Enter the User id of the unauthenticated user. This user will be used when protected resources are included in the Not-Enforced List. The Agent Interceptor returns this as the user to be used for accessing the protected resources included in the Not Enforced List.

Enable Console Integration: Select this to enable console level integration of

Sun ONE Identity Server with WebSphere Administrative Console.

Enable Not-Enforced List Cache: Select this item to enable caching of Not-Enforced List evaluation results.

Number of Entries in Cache: Specify the number of entries that cache can hold at a given instance.

Cache Expiration Time: Specify the time in seconds to be used as the limit for entries that can exist in the Not-Enforced List cache.

For more details on each of these items, see “Agent Configuration Details,” on page 214.”

10. When displayed, review the summary of the installation information you have specified. Press Enter to continue. The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation
```

11. Type 1 to start the installation. At then end of the installation, the following text is displayed:

```
Installation details:
Product Result More Info

1.Sun_TM_ONE Identity Server Policy Agent Installed Available
2.Done

Enter the number corresponding to the desired selection for more
information, or enter 2 to continue [2] {"!" exits}:
```

12. To see log information, type 1. To exit the Installation program, type 2.

13. Start WebSphere Application Server using the following command:

```
# WAS_root_dir/bin/startupServer.sh
```

Configuring WebSphere Application Server

After the Sun ONE Identity Server Policy Agent for WebSphere has been installed on your system, Web Trust Association must be enabled for the Agent Interceptor to work appropriately. This can be done from the WebSphere Administrative Console.

1. Start the WebSphere Administrative console using the following command:

```
# WAS_root_dir/bin/adminclient.sh
```

2. In the Administrative Console, choose Console > Security Center.

3. In the Security Center window, click on Authentication Tab.

4. Select the check box Enable Web trust association.

Figure 8-7 Enabling Web Trust Association

Security Center

General **Authentication** **Role Mapping** **Run As Role Mapping** **Administrative Role**

Authentication Mechanism: ☐ Local Operating System ☒ **Lightweight Third Party Authentication (LTPA)**

LTPA Settings

* **Token Expiration:** 120 minutes

☒ **Enable Single Sign On (SSO)**

* **Domain:** eng.siroe.com

☐ **Limit to SSL connections only**

☒ **Enable Web trust association**

Generate Keys... **Import Key...** **Export Key...**

☐ **LDAP** ☒ **Custom User Registry**

Custom User Registry Settings

A Custom User Registry is a user defined registry. Custom user registries are defined by implementing the com.ibm.websphere.security.CustomRegistry interface. The CustomRegistry interface and a sample implementation can be found in the security section of the WebSphere documentation.

* **Security Server ID:** amldapuser

* **Security Server Password:** *****

* **Custom User Registry Classname** gent.websphere.realm.AgentRealm

Special Custom Settings

OK **Cancel** **Apply** **Help**

5. Click Apply. A message "Changes will take effect only after Administration Server is restarted" will be displayed.
6. In the Security Center window, click on Apply and then OK to apply the changes made.
7. Stop all the Application Server instances and also the Administration Server.
8. Restart Administration Server and Application Server instances for changes to take effect.

Application Configuration

The Agent Realm (Custom Registry) component of Identity Server Policy Agent for WebSphere provides runtime mapping of various principals in Identity Server.

Abstract security role names are used by the hosted application in order to determine if the currently authenticated user is authorized to access a particular resource, or is otherwise a member of a given role. This runtime evaluation can occur only if the user is authenticated as an Identity Server principal using Identity Server's authentication service. Without the user being authenticated appropriately, the results of such evaluations done by the Agent Realm will always be negative, resulting in access being denied to the user for the requested resource.

Creating Role-to-Principal Mappings

Once the application has been secured and the Web Trust Association has been enabled, the Trust Association Interceptor intercepts the requests for the application, validates SSOToken, and authenticates to the Security Service of the Application Server, thereby enabling the Agent Realm to successfully resolve the role-to-principal mappings. However, these mappings must first be created in order that the hosted application may use them during runtime.

The following are two ways to create such mappings:

Editing the WebSphere Server specific deployment descriptors

The WebSphere Server specific deployment descriptors may be edited to create the role to principal mappings. These descriptors are in `ibm-application-bnd.xml` file. Refer to the WebSphere Server reference documentation for details on how these descriptors may be edited to create the role to principal mappings. Alternatively, you can refer the information provided in Appendix C for sample descriptors, to create such mappings.

Using the WebSphere Server Administrative Console

The WebSphere Server Administrative console allows you to create the role to principal mappings by editing the deployed application's deployment descriptors. In order to use this facility, the application must be deployed and the WebSphere Server must be up and running at that time. Refer to the WebSphere Server documentation on how to use the Administrative Console to create such mappings for deployed applications.

Application-Specific Agent Configuration

Oftentimes, the deployed applications are partitioned into public and protected parts, which have varying access restrictions. In most cases, the public portions of the application are accessible to unauthenticated users, whereas the protected portions of the application are accessible only to registered users. The Agent can be configured to allow this type of access by letting unauthenticated users access the public portions of the application without requiring that they authenticate using Identity Server's authentication service. This information is provided as a part of the Agent's configuration properties file that is present in the following location:

Agent_Install_Dir/SUNWam/wasAgent/amAgent/config/AMAgent.properties

This file may be edited to provide general as well as application-specific configuration for the Agent.

NOTE The properties specified in the *AMAgent.properties* file are required for the Agent to function properly. Invalid values specified in this file can lead to malfunction of the Agent, application becoming inaccessible, or the entire system becomes unusable. It is recommended that you use extreme caution when modifying the values in this file. Always create backups before such modifications to ensure that you can back-out your changes to restore the system to its original state.

The properties specified in the *AMAgent.properties* file are loaded during the WebSphere Server startup time. Any changes made to this file while the WebSphere Server is running will not take effect until the server has been restarted.

Providing Application-Specific Not-Enforced List

In order to allow unauthenticated users to access portions of the protected application, the *AMAgent.properties* file must include an entry for the specific application. Access to these entry points will be made as the unauthenticated user. The entry that specifies this property is called the application-specific not-enforced list and is identified by the string:

`com.sun.am.policy.config.filter.AppName.notEnforcedList[index]=pattern`

This property requires that you format it correctly in order that it can be used by the Agent during runtime. The entries appearing in italics within this string should be replaced by their appropriate values as follows:

AppName: This string should be replaced by the deployed application's context path without its leading forward-slash character. The context path is the first URI segment that is used to identify which application the user is trying to access. For example, if the user accesses your application by typing the URL:

`http://myserver.mydomain.com/SomeApp/index.html` or

`http://myserver.mydomain.com/SomeApp/SomeModule/doSomething.jsp`

then in both these cases, the AppName is *SomeApp*.

index: This is an integer value starting from 0 for every deployed application and is unique for every entry in the application not-enforced list. For example, the following are two entries of application not-enforced list for an application with the context path */SomeApp*:

```
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[0]=/SomeApp/public/*
```

```
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[1]=/SomeApp/images/*
```

pattern: This is a pattern string that will be matched with the incoming request to evaluate if the request should be allowed to pass without enforcing authentication or not. The pattern string could be a specific URI, for example */SomeApp/public/RegistrationServlet*, or could be a generic pattern using the wild-card character *'*'* that can be used for denoting 0 or more characters in the request URI, for example */SomeApp/public/** will match with any URI that begins with */SomeApp/public/*.

Using this property, you can specify none or many pattern strings and URIs that the Agent will treat as not-enforced. The user requests that match these particular patterns will be allowed to pass through without requiring a valid SSO token in the request.

Special Case: Default Web Application

A default web application on WebSphere Server is accessible without providing any context path in the request URI. For example, the following URL is that of a default web application:

`http://myserver.mydomain.com/index.html`

The above URL does not have an associated context path.

For such applications, the Agent provides a convenient means of identifying that an entry is specific to the default web application. This is done in two steps as follows:

The following property is set to a name that represents the default web application:

```
com.sun.am.policy.config.filter.defaultWebAppName= DefaultWebApp
```

This name is used to specify the application not-enforced list.

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[0]=/index.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]=about.html
```

Using this scheme, the default web application that does not have a context path associated with it, may be configured just like any other application that has a context path. The same rules apply to the default web application for specifying the not-enforced list entries as applicable to the rest of the application. However, the only difference is that the not-enforced list entries cannot begin with the `/DefaultWebApp/` path segment since such a path segment does not exist on the application server in reality. The `AppName` in this case where the actual context path is an empty string, is only provided as a convenience to specify the properties associated with the default web application and should not be used in specifying their values.

Global Agent Configuration

The `AMAgent.properties` file provides a way to specify a global not-enforced list, which will be applicable to all the protected applications that are deployed on the server. This list is specified using the following property:

```
com.sun.am.policy.config.filter.global.notEnforcedList[index]=pattern
```

`index` is an integer value starting from 0 and unique for every entry.

`pattern` is either an exact URI or a pattern specified by using the wild-card character `*` that can be substituted for zero or more characters in the request URI.

Not-Enforced List Usage Considerations

Although the use of Not-Enforced List can be extremely helpful in partitioning your application for public and protected domains, it can also lead to undesirable effects if not used appropriately.

For example, if a request URI that represents a Servlet is matched by some not-enforced list pattern, then the Agent Interceptor will not perform authentication to the Application Server, for users who try to access that particular Servlet. However, consider the case where this Servlet accesses an Enterprise JavaBeans component that is protected by the Agent using role to principal mapping. In such a case, since the user is unauthenticated, the access to the protected component will result in a security violation exception being generated by the application server. Therefore, before an entry is added to the not-enforced list, it must be ensured that the entry does not in any way cover a resource to which the Unauthenticated user has no access, or may try to access such a resource.

Agent Configuration

The core configuration needed by the Sun ONE Identity Server Policy Agent for WebSphere is provided in the `AMAgent.properties` file located in the following directory:

Agent_Install_Dir/wasAgent/amAgent/config

This property file provides many configuration settings that can be modified in order to customize the Agent's operation for your deployment scenario.

NOTE Before proceeding, you must note that this file and the information within the file are critical for the operation of the Agent. It is strongly recommended that you always create a backup of this file before modifying it. Also, it is strongly recommended that you do not modify this file unless it is absolutely necessary. Any invalid data entries present in this file can lead to the malfunction of the Agent, malfunction of the deployed applications, and could render the entire system unusable.

The settings provided in this file can be classified into the following categories:

- Common Configuration
- Audit Configuration
- Realm Configuration
- Global Interceptor Configuration
- Application Interceptor Configuration
- Debug Engine Configuration

The following sections detail the settings for each of these classifications.

Common Configuration

Settings in this section are general settings that affect the behavior of the Agent as a whole.

Organization Name

Key: com.sun.am.policy.config.org

Description: This property specifies the organization name to be used when searching for principals in Sun ONE Identity Server.

Valid Values: String representing the organization name in Sun ONE Identity Server. This property is set during Agent installation and must not be changed unless absolutely necessary.

Example: com.sun.am.policy.config.org=dc=sun,dc=com

Root Suffix

Key: com.sun.am.policy.config.rootsuffix

Description: This property specifies the root suffix to be used when searching for principals in Sun ONE Identity Server.

Valid Values: String representing the root suffix in Sun ONE Identity Server. This property is set during Agent installation and must not be changed unless absolutely necessary.

Example: com.sun.am.policy.config.rootsuffix=dc=sun,dc=com

People Container Level

Key: com.sun.am.policy.config.realm.peopleContainerLevel

Description: This property specifies the people container level to be used when searching for principals in Sun ONE Identity Server.

Valid Values: Non-zero unsigned integer representing the People Container Level in Identity Server, which may be used when searching for principals. This property is set during Agent installation and must not be changed unless absolutely necessary.

Example: com.sun.am.policy.config.realm.peopleContainerLevel=1

Audit Configuration

These settings are exclusively used to configure the Audit Engine used by the Agent.

Language Code

Key: `com.sun.am.policy.config.audit.localeLanguageCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeLanguageCode` must be a valid ISO Language Code. The default value of this property is `en`.

Example: `com.sun.am.policy.config.audit.localeLanguageCode=en`

NOTE For more information, refer to ISO 639 specification at
<http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt>

Country Code

Key: `com.sun.am.policy.config.audit.localeCountryCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeCountryCode` must be a valid ISO Country Code. The default value of this property is `US`.

Example: `com.sun.am.policy.config.audit.localeCountryCode=US`

NOTE For more information, refer to the ISO 3166 specification:
http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html

Audit Log File

Key: `com.sun.am.policy.config.audit.logfile.name`

Description: This property specifies the Audit log file to be used for recording Audit messages.

Valid Values: String representing the complete path name of the file to be used by the Agent to record Audit messages.

Example: `com.sun.am.policy.config.audit.logfile.name=/audit/agent.log`

NOTE Ensure that the WebSphere Server process has sufficient permissions to write to this file.

Specifying an invalid value specified for this property may result in the failure of the system to start up correctly.

Audit Log File Rotation Flag

Key: `com.sun.am.policy.config.audit.logfile.rotate`

Description: This property specifies if the Audit log file should be rotated by the Agent.

Valid Values: true/false. The default value of this property is false and should be changed as necessary.

Example: `com.sun.am.policy.config.audit.logfile.rotate=false`

Audit Log File Rotation Size

Key: `com.sun.am.policy.config.audit.logfile.rotate.size`

Description: This property specifies the approximate size of the Audit log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated. The default value of this property is 52428800 bytes (~ 50 MB) and should be changed as required.

Example: `com.sun.am.policy.config.audit.logfile.rotate.size=52428800`

Realm Configuration

These settings are used to configure the Agent Realm component.

The Agent caches the SSOToken for the logged in user and makes it available for the application through the AgentUser APIs. The cache behavior is controlled by the Realm Configuration properties in the `AMAgent.properties` file.

Realm Name

Key: `com.sun.am.policy.config.realm.name`

Description: Used by the Realm implementation to identify itself to the WebSphere Application Server. Any changes to this should be accompanied by the appropriate change in WebSphere Administrative Console, Security Center and property file `sas.client.props`.

Allow Console Integration Flag

Key: `com.sun.amagent.config.websphere.allowConsoleIntegration`

Description: This property specifies if the Agent Realm should allow console level integration of Identity Server with WebSphere Server.

Valid Values: `true/false`

Example: `com.sun.amagent.config.websphere.allowConsoleIntegration=true`

When set, the result of console level integration is that the WebSphere Server Administrator will be able to see the list of Identity Server principals in WebSphere Server console.

This setting enables the WebSphere Server Administrator to see the Identity Server principals in WebSphere Server console, it should be used with caution.

The default value of this setting is false and should be changed as necessary.

SSO Cache Cleanup Size

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupSize`

Description: This property specifies the maximum number of entries in the SSO Cache maintained by the Agent Realm after which a cleanup will be initiated.

Valid Values: Any positive integer value indicating the number of entries in the Agent Realm's SSO Cache which when exceeded will initiate a clean up operation to ensure minimal and appropriate use of the system's memory.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance and/or result in the loss of availability of SSO token in the system.
- The value for optimal system performance will depend on the type of application deployed, the number of active user sessions that the application is subject to during peak periods, and the average user session time value.
- To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
- This property defaults to a value of 1000.

Example: `com.sun.am.policy.config.realm.ssoCacheCleanupSize = 1000`

SSO Cache Cleanup Lock Time

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime`

Description: This property specifies the amount of time in seconds that the Agent Realm will wait before initiating the next cleanup process for the SSO cache if the last cleanup process did not result in freeing any memory.

Valid Values: Any positive integer value indicating the number of seconds that the cleanup process will not be restarted if the last cleanup process did not result in freeing sufficient memory.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
- The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
- To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
- This property defaults to a value of 1200.

Example: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime = 1200`

SSO Cache Cleanup Bound Size

Key: com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize

Description: This property specifies the maximum number of entries in the Agent Realm SSO Cache that will be inspected during cleanup process. This value when set appropriately will result in overall improvement of the response time of the system when it undergoes a cache cleanup operation.

Valid Values: Any positive integer value indicating the number of entries the cleanup process will inspect in the SSO cache during the cleanup operation. This value can be set independently with respect to the value of the property com.sun.am.policy.config.realm.ssoCacheCleanupSize.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
 - The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 50.
-

Example: com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize = 50

Global Interceptor Configuration

These settings are used to configure the Agent Interceptor component.

IsOnly Interceptor Flag

Key: com.sun.am.policy.config.filter.isOnlyInterceptor

Description: This property indicates if this is the lone interceptor configured in the system. By default, this is set to true. If additional interceptors need to be added to WebSphere, then this should be set to false and appropriate changes should be made in the file trustedServers.properties.

Valid Values: true/false. The default value is true.

If set to true, the absence of SSOToken in the request will cause the request to get denied. If set to false, the absence of SSOToken will result in control being passed on to the next interceptor that is installed.

UnAuthenticated User

Key: com.sun.am.policy.config.filter.notEnforcedList.nonAuthUser

Description: This unauthenticated user is used to access the resources specified in the Not-Enforced List. Access to the resources in the Not-Enforced List will be successful, only if the user has the required roles for accessing them.

Valid Values: Any valid user in the Identity Server.

SSO Token Name

Key: com.sun.am.policy.config.filter.ssoTokenName

Description: This property specifies the name of the cookie that represents the SSO Token.

Valid Values: A string that represents the name of SSO Token cookie issued by the Sun ONE Identity Server authentication service. This property is set during Agent installation and need not be changed unless absolutely necessary.

Example: com.sun.am.policy.config.filter.ssoTokenName=iPlanetDirectoryPro

Not-Enforced-List Cache Enable Flag

Key: com.sun.am.policy.config.filter.notEnforcedList.cache

Description: This property specifies if the requested URIs that are evaluated as enforced or not-enforced may be cached to increase the performance of the system.

Valid Values: true/false. The default value of this property is true.

Example: com.sun.am.policy.config.filter.notEnforcedList.cache=true

Not-Enforced-List Cache Size

Key: com.sun.am.policy.config.filter.notEnforcedList.cacheSize

Description: This property specifies the number of entries that will be kept in the cache of not-enforced URIs and enforced URIs by the Agent.

Valid Values: Non-zero unsigned integer indicating the number of enforced as not enforced request URIs to be cached during runtime.

NOTE Values that are valid but not suited for the deployment scenario may result in degradation of system performance.

The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries and a host of other deployment specific factors.

To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.

Example: `com.sun.am.policy.config.filter.notEnforcedList.cacheSize=1000`

Not-Enforced-List Cache Expiration Time

Key: `com.sun.am.policy.config.filter.notEnforcedList.cacheTime`

Description: This property specifies the amount of time in seconds that will be used to evaluate if a cached entry can be removed from the cache to free up resources for new cache entries.

Valid Values: Non-zero unsigned integer indicating the time in seconds that will be used as the cache expiration time for entries in the cache during a cleanup operation.

NOTE Values that are valid but not suited for the deployment scenario may result in degradation of system performance.

The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries, and a host of other deployment specific factors.

To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.

Example: `com.sun.am.policy.config.filter.notEnforcedList.cacheTime=60`

SSO Token URL Decode Flag

Key: `com.sun.am.policy.config.filter.urlDecodeSSOToken`

Description: This property indicates if the SSOToken needs to be URL Decoded by the Agent before it may be used.

Valid Values: true/false. The default value of this property is set to true.

NOTE Valid, but inappropriate value of this property may lead to the application being unreachable by the users.

Example: com.sun.am.policy.config.filter.urlDecodeSSOToken=true

Default Web Application Name

Key: com.sun.am.policy.config.filter.defaultWebAppName

Description: This property specifies a name for the Default Web Application deployed on the application server.

Valid Values: A string consisting of lower case and or upper case letters that can be used as the name for the default web application. The default value of this property is DefaultWebApp.

NOTE This property is necessary if the protected application is deployed as the default web application.

Example: com.sun.am.policy.config.filter.defaultWebAppName=DefaultWebApp

Global Not-Enforced List

Key: com.sun.am.policy.config.filter.global.notEnforcedList[index]

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters. The syntax of this property is as follows:

com.sun.am.policy.config.filter.global.notEnforcedList[index]=pattern
index is an integer that starts from 0 and increments for every entry in this property
list pattern is a string that represents request URIs that are not enforced by Agent.

The pattern string may consist of wild-card character '*', which may match zero or more characters.

The index must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries.

NOTE No value for this property indicates an empty global not-enforced list.

Example:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=*.gif
com.sun.am.policy.config.filter.global.notEnforcedList[1]= public/*
com.sun.am.policy.config.filter.global.notEnforcedList[2]= /images/*
```

Application Interceptor Configuration

These settings are used to configure the Agent Interceptor for a particular application.

Application Not-Enforced-List

Key: com.sun.am.policy.config.filter.AppName.notEnforcedList[index]

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent for a particular application. All the URIs that match any of these patterns will be accessed as Unauthenticated User.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters. The syntax of this property is as follows:

```
com.sun.am.policy.config.filter.AppName.notEnforcedList[index]= pattern
```

AppName is the context path name without the leading forward-slash character (/) for the deployed application.

index is an integer that starts from 0 and increments for every specified property for the particular application.

pattern is a string that represents the URIs that are not enforced by the Agent.

NOTE The pattern string may consist of wild-card character '*', which may match zero or more characters.

The index must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries. The index value will be independent for different AppNames specified in this property list.

If the protected application is the default web application, the AppName should be set to the same string as specified for the value of the property Default Web Application Name.

No value for this property indicates an empty not enforced list.

Example:

```
com.sun.am.policy.config.filter.Portal.notEnforcedList[0]= /Portal/GuestPages/*
com.sun.am.policy.config.filter.Portal.notEnforcedList[1]= /Portal/Registration/*
com.sun.am.policy.config.filter.Portal.notEnforcedList[2]=
/Portal/WebServices/PollServlet
com.sun.am.policy.config.filter.BankApp.notEnforcedList[0]=
/BankApp/ModuleGuestTour/*
com.sun.am.policy.config.filter.BankApp.notEnforcedList[1]=
/BankApp/index.html
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[0]=/index.html
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]= /about.html
```

Debug Engine Configuration

These settings are used to configure the Debug Engine to generate diagnostic information.

Debug Level

Key: com.sun.am.policy.config.debug.level

Description: This property specifies the amount of debug messages that will be generated by the Agent's Debug Engine. This property works only when a Debug Log File location is specified.

Valid Values: Any of 0, 1, 3, 7, 15, and 31. These values indicate the following:

0 = No debugging

1 = Only Error messages

3 = Error and Warning messages

7 = Error, Warning and Brief Informational messages

15 = Error, Warning and Verbose Informational messages

31 = Error, Warning and Very Verbose Informational messages

NOTE For better performance of the system, this property should be set to a value 0. Values other than 0 will affect the system performance depending upon the amount of information the Debug Engine has to generate.

Values other than the ones specified in the Valid Values list above will lead to invalid configuration of the Debug Engine, thereby affecting the volume of messages that will be generated.

Example: `com.sun.am.policy.config.debug.level=7`

Debug Log File

Key: `com.sun.am.policy.config.debug.logfile.name`

Description: This property specifies the Debug log file to be used for recording Debug messages.

Valid Values: String representing the complete path name of the file to be used by the Agent to record Debug messages.

NOTE Ensure that the WebSphere Application Server process has sufficient permissions to write to this file.

Invalid or empty value of this property will lead to Debug messages not being logged in the log file.

Example: `com.sun.am.policy.config.debug.logfile.name=/debug/agent_debug.log`

Debug Log File Rotation Flag

Key: `com.sun.am.policy.config.debug.logfile.rotate`

Description: This property specifies if the Debug log file should be rotated by the Agent.

Valid Values: true/false. The default value of this property is false and should be changed as necessary.

Example: com.sun.am.policy.config.debug.logfile.rotate=false

Debug Log File Rotation Size

Key: com.sun.am.policy.config.debug.logfile.rotate.size

Description: This property specifies the approximate size of the Debug log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated. Default value of this property is 52428800 bytes (~ 50 MB) and should be changed as required.

NOTE This property is not used if the Debug Log File Rotation Flag is set to false.

Example: com.sun.am.policy.config.debug.logfile.rotate.size=52428800

Debug Time/Date Format String

Key: com.sun.am.policy.config.debug.date.format

Description: This property specifies the format of time stamp that is used to mark the exact time when the Debug message was recorded.

Valid Values: Valid java.text.SimpleDateFormat Time Format Syntax string. For more information, refer to:

<http://java.sun.com/j2se/1.3/docs/api/java/text/SimpleDateFormat.html>

NOTE The default value of this property is set to <MMM d, yyyy h:mm:ss a z> 'Agent' and should be changed as necessary.

Invalid value of this property will result in loss of time stamp data with debug messages.

Example: com.sun.am.policy.config.debug.date.format=[yyyy/MM/dd HH:mm:ss zzz]

Debug Print STDOUT Flag

Key: com.sun.am.policy.config.debug.print.stdout

Description: This property specifies if the Debug Engine should print the debug messages on Standard Output stream.

Valid Values: true/false. The default value of this property is true and should be changed as necessary.

Example: com.sun.am.policy.config.debug.print.stdout=true

Using Agent and Sun ONE Identity Server SDK APIs

You can use the Sun ONE Identity Server SDK APIs to create security and identity aware applications. Such applications can perform custom security and identity related tasks such as application level policy enforcement by exploiting the rich security and policy infrastructure offered by the Sun ONE Identity Server. When you install the Sun ONE Identity Server Policy Agent for WebSphere Application Server, the Sun ONE Identity Server SDK becomes available for your application to use.

While the availability of the SDK in itself is sufficient for the developers of the application to make it security aware, the Policy Agent for WebSphere Application Server further facilitates this by ensuring that the Single Sign-On (SSO) Token is available for the logged on user who at any given point in time may be using the deployed system.

When the logged on user accesses a protected resource, the Agent Interceptor ensures that the user be appropriately authenticated and that the corresponding Principal be available throughout the system. The Principal instance may be accessed by the J2EE programmatic security calls such as `HttpServletRequest.getUserPrincipal()` and `EJBContext.getCallerPrincipal()`. The Principal instance returned by these methods represent the user as authenticated by the Sun ONE Identity Server's authentication service. The `AgentUser` API provided by the Policy Agent can be used to access the user's SSO Token from anywhere within the application in the following simple steps:

1. Create an instance of the `AgentUser`

- a. From `HttpServletRequest`

```
import com.sun.amagent.websphere.realm.AgentUser;  
import com.iplanet.amagent.arch.AgentException;  
  
.....  
.....
```

```

AgentUser user=null;
HttpServletRequest request = getHttpRequest();

try {
    user = new AgentUser(request);
}
catch (AgentException ex) {
    throw new RuntimeException("Error in creating the instance of
AgentUser !!!");
}

```

b. From EJBContext

```

import com.sun.amagent.websphere.realm.AgentUser;
import com.ipplanet.amagent.arch.AgentException;

.....
.....
AgentUser user=null;

EJBContext context = getEJBContext( )

try {
    user = new AgentUser(context);
}
catch (AgentException ex) {
    throw new RuntimeException("Error in creating the instance of
AgentUser !!!");
}

```

- 2. Use the AgentUser API to retrieve the SSO Token associated with the current principal. For example:**

```

...

String ssoTokenId = null;

if (user != null) {
    ssoTokenId = user.getSSOTokenID();
}

...

```

Once the SSO Token string is acquired, it can be used for subsequent calls into the Identity Server SDK APIs.

NOTE The Agent uses these configuration settings which ensure the availability of the SSO token associated with the user in the Agent Realm. If the values for these settings is not appropriate for your deployment scenario, it may result in the degradation of the overall system performance. See "“Realm Configuration” on page 231."

SSL Configuration

The SSL Configuration involves the following tasks:

- Configuring SSL for the Web Server

Configuring SSL for the Web server depends on the Web server used. Refer to the appropriate Web Server documentation.

- Configuring SSL for WebSphere plug-ins for Web servers.

See the WebSphere documentation at:

<http://www-3.ibm.com/software/webervers/appserv/doc/v40/ae/info-center/was/06061801a07.html#pi>

- Configuring SSL for WebSphere Application Server.

See the documentation at

<http://www-3.ibm.com/software/webervers/appserv/doc/v40/ae/info-center/was/06061801a07.html#was>

Configuration Changes in AmConfig.properties

This is the additional step to be performed when the Identity Server is running in SSL Mode. The `AmConfig.properties` file is located at `Agent_Install_Dir/SUNWam/wasAgent/amSDK/lib`.

```
com.ipplanet.am.server.protocol=https
```

```
com.ipplanet.am.console.protocol=https
```

```
com.ipplanet.am.naming.url=https://<IS_HOST>:<IS_PORT>/amserver/naming-service
```

```
com.ipplanet.am.notification.url=https://<IS_HOST>:<IS_PORT>/amserver/notification-service
```

Additionally make the following changes if the Directory Server is configured for LDAPS.

```
com.ipplanet.am.directory.ssl.enabled=true
com.ipplanet.am.directory.port=<LDAPS Port>
```

Configuration Changes in serverconfig.xml

If the Directory Server is configured for LDAPS update the port and type information in serverconfig.xml. This file is located at

Agent_Install_Dir/SUNWam/wasAgent/amSDK/config/ums.

```
<iPlanetDataAccessLayer>
<ServerGroup name="default" minConnPool="1" maxConnPool="10">
<Server name="Server1" host="sleeper.india.sun.com" port="<LDAPS
Port>" type="SSL" />
.....
.....
.....
.....
</ServerGroup>
</iPlanetDataAccessLayer>
```

Importing the root CA cert into Application Server JVM Keystore

Export the root CA certificate for the CA (who signed the certificate used by the Identity Server) into a Base64-encoded ASCII data file /tmp/cacert.txt

Import this cert into AppServer JVM Keystore as follows:

```
# WAS_root_dir/java/bin/keytool -import -trustcacerts -alias cmscacert
-keystore ../jre/lib/security/cacerts -file /tmp/cacert.txt
```

Enter keystore password: changeit

Trust this certificate? [no]: yes

keystore: all cert installed

Verify that Cert is added

```
# WAS_root_dir/java/bin/keytool -list -keystore
WAS_root_dir/java/jre/lib/security/cacerts
```

Enter keystore password: changeit

Uninstalling the Agent

When you install the Identity Server Policy Agent for WebSphere Application Server software, an uninstallation program is created in the installation directory. Using this uninstallation program the Agent can be removed completely from your system. While the uninstallation program deletes all the installed files from your system, certain files such as audit log messages are not deleted. You can delete them manually.

Launching the Uninstallation Program

The uninstallation program for Solaris platform may be launched by executing the generated `uninstall_wasagent` script located in the installation directory. The following steps provide details on how to achieve this:

1. Login as root.
2. Go to *Agent_Install_Dir*.
3. Set your `JAVA_HOME` environment variable to the JDK supplied with WebSphere Server 4.0.4:

```
WAS_root_dir/ java
```

The uninstallation program provides two types of interfaces, a graphical user interface (GUI) and a command-line interface. In most cases, the GUI installation program can be used for uninstalling the Agent. However, in cases when you are uninstalling the Agent over a telnet session on a remote server and do not have windowing capabilities, then the GUI uninstallation program cannot be used. In such a case it is recommended that you use the command line uninstallation program for uninstalling the Agent. This can be launched by executing the `uninstall` script and passing in a command line argument `-nodisplay` as follows:

```
#./uninstall_wasagent -nodisplay
```

However, if you choose to use the GUI uninstallation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI uninstallation program window appears on the correct console.

NOTE If you choose to use the command line uninstallation program using the `-nodisplay` option, you may skip the next step and proceed directly to the next section, which details out the uninstallation procedure.

1. Launch the GUI uninstallation program by invoking the uninstall script as follows:

```
# ./uninstall_wasagent
```

The uninstallation program requires that you setup your JAVA_HOME variable correctly as pointed out in the Step 3. However, in case you have set the JAVA_HOME variable incorrectly, the uninstall script will prompt you for supplying the correct JAVA_HOME value.

```
Enter JAVA_HOME location (Enter "." to abort):
```

2. Type the full path to the JDK installation directory for launching the installation program. Otherwise, enter a period (.) to abort the installation

In order that the GUI uninstallation program be displayed on your console, the DISPLAY environment variable of your shell must be set correctly. In case your DISPLAY environment variable is not set at the time of invoking the uninstall script, the uninstallation program will prompt you for the DISPLAY environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

3. Provide the DISPLAY value to the installation program by typing in the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation at this time.

Uninstalling the Agent Using GUI

The uninstallation program begins with a Welcome screen.

1. Click Next to step through the uninstallation screens, answering the questions.
2. In the Ready to Uninstall screen, review the uninstallation information. If you need to make changes, click Back. Otherwise, click Uninstall Now.
3. The Uninstall Progress screen displays the progress of uninstall process.
4. In the Uninstallation Summary window, click Details for a detailed summary of the configuration information that was processed during uninstallation. Click Exit to end the program

NOTE It is recommended that you check the log file by clicking the Details button to identify errors. If you find errors, try to uninstall the Agent manually by removing the package SUNWamwas using the `pkgrm` command.

5. Remove the Realm Administrator from the AdminRole.

Uninstalling the Agent Using Command Line

1. From the directory where the agent is installed, enter the following command at the command line:

```
# ./uninstall_wasagent -nodisplay
```

2. When prompted, Please enter path to pick up java: Type the full path where WebSphere JDK is located.
3. At the following prompt, type 1 to begin uninstallation.

```
Ready to Uninstall
1. Uninstall Now
2. Start Over
3. Exit Uninstallation
```

4. At the following prompt, to see log information on the agent, enter 1. To exit the Uninstallation program, enter 2.

```
Product Result More Info
1. Sun_TM_ONE Identity Server Policy Agent Details Available
2. Done
```


Troubleshooting Information

Installation/Uninstallation Problems

Server Startup Error due to Agent Realm (Custom Registry) initialization failure, after the Agent installation

The realm initialization fails when the Realm is unable to authenticate to the Identity Server. This may be due to one of the following reasons.

Incorrect values were given for amldapuser password. You can try giving the correct values in the Administrative Console Security Center. Apply the changes and restart WebSphere.

Installation Program fails to add System Property for Application Server Instances, for the Identity Server SDK config directory (com.ipplanet.coreservices.config). You can add this property

`com.ipplanet.coreservices.config=Agent_Install_Dir/SUNWam/wasAgent/amSDK/config/ums` to each Application Server instance, under JVM Settings in the Administrative Console. Again save the changes and restart WebSphere.

If the Identity Server is running in SSL Mode, then verify the following.

The certificate for the CA that issued the Certificate for the Identity Server, is imported into the KeyStore of the JVM used by the WebSphere. See "SSL Configuration."

The System property `java.protocol.handler.pkgs` was added by the Agent installation program to each Application Server instances. This can be found under JVM Settings for each Application server instance, in the WebSphere Administrative console. This property must be set to the value:

`com.ibm.net.ssl.internal.www.protocol`

Verify the Identity Server configuration parameters in the `AMConfig.properties` file.

Access to the resource fails even after Authenticating to the Identity Server.

If you are accessing a protected resource from an unprotected resource, make sure that you have security constraint (dummy) defined for the unprotected resource with the role mapped to any principal other than the special subject `Everyone`.

Make sure that the Web trust association is enabled, and WebSphere was restarted after enabling Web Trust Association.

Make sure that the following lines are present in the trusted

WAS_root_dir/properties/trustedservers.properties

```
com.ibm.websphere.security.trustassociation.enabled=true
com.ibm.websphere.security.trustassociation.types=amagent
com.ibm.websphere.security.trustassociation.amagent.interceptor=
com.sun.amagent.websphere.interceptor.AgentInterceptor
com.ibm.websphere.security.trustassociation.amagent.config=AMAge
nt
```

Fixing the Problems Manually

During the installation, the Agent installation program performs modifications for certain existing files on your system. These modifications are undone by the uninstallation program. The backups of these files are created before the modifications are made. In case you encounter any failures during the install it is possible to bring the system back to its original state manually by restoring the backed up files. These backed up files are copied under

Agent_Install_Dir/SUNWam/wasAgent/backup directory. This directory is deleted if the uninstallation was completed successfully.

The following files are modified during installation and the modifications are undone during the uninstallation of the Agent:

Admin Server Configuration file:

WAS_root_dir/bin/admin.config

Trusted Servers file:

WAS_root_dir/properties/trustedservers.properties

Client Configuration file:

WAS_root_dir/properties/sas.client.props

Agent uninstallation also creates a backup of the version of the above files, just before the uninstallation under the directory *WAS_root_dir*/amAgentBackup. In case of any errors during uninstallation these can be used to manually reapply those changes to the current copy of these files.

Policy Agent for Sun ONE Application Server 7.0

This chapter describes how to install and configure Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0. Topics include:

- Supported Platforms
- Guidelines
- Installing the Agent
- Application Configuration
- Agent Configuration
- Uninstalling the Agent

Supported Platforms

The Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0 is supported on the following platforms:

- Solaris 8
- Solaris 9
- Windows 2000 Server

Guidelines

In order to use the Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0 in the most optimal manner, it is recommended that you follow the following guidelines:

- **Use Agent-Based Authentication**

After the Agent is installed and the application has been configured to use Agent Filter component, the Agent Filter component enforces authentication for all web based access to the protected application's enforced portions. Working in tandem with the Agent Realm component, the Agent Filter ensures that the J2EE policies defined for the protected application get evaluated correctly based on the set role-to-principal mappings, at the same time offering other key services such as Single Sign-On (SSO). Therefore, it is recommended that protected applications do not use their own authentication mechanism or any container based authentication mechanisms which would result in the Agent Filter component being bypassed during the application operation.

- **Create Enhanced Security Aware Applications**

The Agent provides the rich APIs offered by Identity Server SDK libraries, which are available for use within the protected application. Using these APIs, the application architect can create enhanced security aware applications that are custom tailored to work in the security framework offered by Identity Server. For more information on how to use the Sun ONE Identity Server SDK, refer to *Sun ONE Identity Server Programmer's Guide*.

Installing the Agent

The Sun ONE Identity Server Agent for Sun ONE Application Server 7.0 can be installed on these platforms—Solaris 8, Solaris 9, or Windows 2000 Server. The following sections provide the detailed steps necessary for installing the Agent on each of these platforms.

Pre-installation Tasks

The following tasks must be completed before the Agent can be installed:

- **Install Sun ONE Application Server 7.0**

Refer Sun ONE Application Server documentation for the necessary details. When the server is installed, test the installation by using the provided sample Applications to ensure that the server is installed correctly.

- **Test the deployment of application that you intend to protect**

Before installing the Agent, it is important that the application that must be protected be deployed and tested for simple functionality. Once it is established the application can be deployed successfully, you are ready to install the Agent.

- **Stop the Application Server instance on which you want to install the Agent.**

Stop the Application Server instance using the following command:

```
/S1AS_Install_Dir/SUNWappserver7/domains/domain-instance/server-instance
/bin/stopserv
```

Launching the Installer on Solaris Platform

The binaries for Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0 for Solaris 8 and Solaris 9 platform is provided as a tar-gzip archive. Copy this archive on the machine where Sun ONE Application Server is installed. Perform the following steps to launch the installation program:

1. Login as `root`.
2. Unzip the binary archive in a convenient location using the following command:


```
# gzip -dc AS70_agent_2.0_sparc-sun-solaris2.8.tar.gz | tar xvf -
```
3. Set your `JAVA_HOME` environment variable to JDK version 1.4.0 or higher. If your system does not have the required version of JDK, use the JDK supplied with Sun ONE Application Server 7.0. This JDK is located under:

```
S1AS_Install_Dir/jdk
```

The installation program provides two types of interfaces—a graphical user interface (GUI) and a command line interface. In most cases, the GUI installer can be used for installing the Agent. However, in cases when you are installing the Agent over a telnet session on a remote server and do not have windowing capabilities, then the GUI program cannot be used. In such a case, it is recommended that you use the command line installation program. This can be launched by executing the `setup` script and passing a command line argument `-nodisplay` as follows:

```
# ./setup -nodisplay
```

However, if you choose to use the GUI program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI installation screens appear on the correct console.

NOTE If you choose to use the command-line program using the `-nodisplay` option, you may skip the following step and proceed directly to the next section, which details out the installation procedure.

4. Launch the GUI installer by invoking the `setup` script as follows:

```
# ./setup
```

The installation program requires that you set up your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, in case you have incorrectly set the `JAVA_HOME` variable, the `setup` script will prompt you for supplying the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the Sun ONE Application Server JDK installation directory for launching the installation program. Otherwise, enter a period (.) to abort the installation.

In order that the GUI installer be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. If your `DISPLAY` environment variable is not set at the time of invoking the `setup` script, the installer will prompt you for the `DISPLAY` environment variable value as follows:

```
Please enter the value of DISPLAY variable (Enter "." to abort):
```

Provide the `DISPLAY` value by typing the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

NOTE You can also use `agent_SunOS.class` file to install the agent. You can find this file in the directory where you have untarred the binaries.

Launching the Installer on Windows 2000 Server

The binaries for Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0 for Windows platform are provided as a zip archive. Copy this archive on to the machine where Application Server server is installed. Perform the following steps to launch the installation program:

1. You must have administrative privileges when you run the installation program. If you do not have administrative privileges, either log on as the “Administrator” user or request such privileges to be granted to your account by the system administrator of the machine or domain as applicable.
2. Unzip the Agent binaries in a convenient location. This may be done by using any Zip utility available on your system.

The above operation results in two executable files `setup.bat` and `setup.exe`, which may be used to launch the installer. Each of these two files provide different features for launching the installer. You can choose either of these two files depending upon your installation requirements.

Using `setup.bat`

In order to use the `setup.bat` file to launch the installer, you must have JDK version 1.4.0 or higher. This can be verified by typing the following command in a command prompt window:

```
C:\> java -version

java version 1.4.0_02

Java(TM) 2 Runtime Environment, Standard Edition (build
1.4.0_02-b03)

Java HotSpot(TM) Client VM (build 1.4.0_02-b03, mixed mode)
```

If you do not have JDK of required version in your system path, you can use the JDK supplied with Application Server 7.0 located at:

```
SIAS_Install_Dir\jdk
```

The `setup.bat` can be executed by typing the file name at the command prompt window in a directory where it is present, or by double clicking the file in Windows Explorer. For example:

```
C:\>setup.bat
```

The installation program provides two types of interfaces—a GUI and a command line interface. By invoking the `setup.bat` file from a command prompt window or by double clicking the file in Windows Explorer, the installation program is launched in the GUI mode. To launch the installation program with the command line argument `-nodisplay`, type the following:

```
C:\>setup.bat -nodisplay
```

Using setup.exe

You can invoke `setup.exe` either from the command prompt window or by double clicking the file from Windows Explorer. Unlike `setup.bat`, `setup.exe` does not take the `-nodisplay` command line argument and can launch only the GUI program.

NOTE	Since the command line installer cannot be launched using <code>setup.exe</code> , in cases where the command line installer is required, it is recommended that you use <code>setup.bat</code> to launch the command line installer.
-------------	---

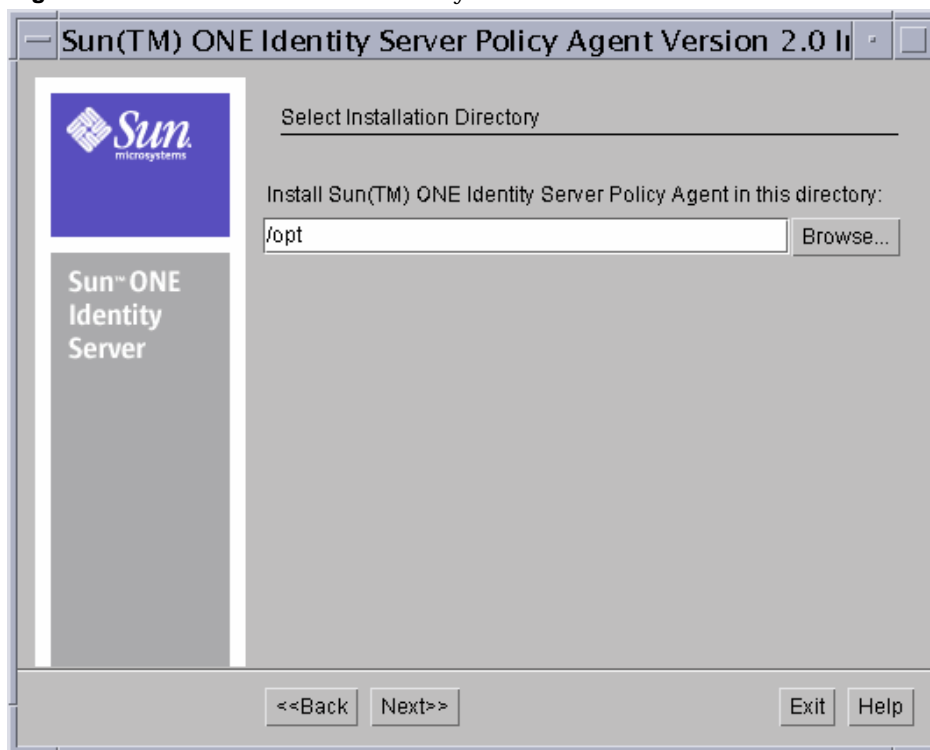
Installing the Agent Using GUI

The installation program begins with a Welcome screen. Click Next to step through the installation screens, answering the questions.

1. Read the License Agreement. Click Yes (Accept License) to continue with the installation.

If you do not accept all the terms of the Software License Agreement, click No to end the Installation program without installing the product.

2. In the Select Installation Directory screen, enter the path where you want to install the Agent.

Figure 9-1 Select Installation Directory screen

If you wish to install the Agent in a directory different from the default installation directory, click the Browse button and choose the directory. Once you have selected the appropriate directory, click the Next button to proceed to the next screen.

NOTE If you select a directory that does not exist on your system, the installation program will prompt you to specify if the new directory should be created. You can either choose to create this new directory by clicking on the Create Directory button or select a new directory by clicking on the Choose Another Directory button.

3. In the Sun ONE Identity Server Information screen, provide the following information about the Sun ONE Identity Server and click Next.

Figure 9-2 Sun ONE Identity Server Information Screen

Sun(TM) ONE Identity Server Information

Enter the server information where the Sun(TM) ONE Identity Server Service is installed.

Sun(TM) ONE Identity Server Host: nila.Eng.Siroe.COM

Sun(TM) ONE Identity Server Port: 58080

Sun(TM) ONE Identity Server Protocol: ☒ http ☐ https

Sun(TM) ONE Identity Server Deployment URI: amserver

amAdmin Password: ****

Re-enter Password: ****

<<Back Next>> Exit Help

Sun ONE Identity Server Host: Enter the fully qualified host name of the system where Sun ONE Identity Server is installed. For example, nila.eng.siroe.com.

Sun ONE Identity Server Port: Enter the port number for the Web Server that runs Sun ONE Identity Server Services. The default port number is 58080.

Sun ONE Identity Server Protocol: Select the protocol that will be used by the Agent to communicate with Sun ONE Identity Server services. If the web server has been configured for SSL, select HTTPS; otherwise select HTTP.

Sun ONE Identity Server Deployment URI: Enter the URI that should be used for accessing Sun ONE Identity Server Services. The default URI is amserver.

amAdmin Password: Enter the password for amAdmin user.

Re-enter Password: Re-enter the password for amAdmin user for confirmation.

NOTE The password supplied during installation is recorded by the Agent in a secure manner. However, if in the future you change this password in Identity Server, you will have to update the password in the Agent as well. This can be done by using the `agentadmin` tool provided with the Agent. Once the Agent has been installed on the system, the `agentadmin` tool can be invoked from the location:
Agent_Install_Dir/SUNWam/asAgent/bin/agentadmin

The `agentadmin` tool is available as `agentadmin.bat` on the Windows platform.

To change the password, invoke this tool as follows:

```
#./agentadmin -password oldpassword newpassword
```

4. In the Directory Server Information screen, provide the following information about the Directory Server that is associated with Sun ONE Identity Server services.

Figure 9-3 Directory Server Information Screen

The screenshot shows a window titled "Sun(TM) ONE Identity Server Policy Agent Version 2.0". On the left is a sidebar with the Sun Microsystems logo and the text "Sun™ ONE Identity Server". The main area is titled "Directory Server Information" and contains the instruction: "Enter the directory server information corresponding to the Sun(TM) ONE Identity Server Services." Below this are four input fields:

Directory Host:	nila.Eng.Siroe.COM
Directory Port:	389
Root Suffix:	dc=Siroe,dc=COM
Installation Organization:	dc=Siroe,dc=COM

At the bottom of the window are four buttons: "<<Back", "Next>>", "Exit", and "Help".

Directory Host: Enter the fully qualified host name of the system where the Directory Server is installed.

Directory Port: Enter the port number used by the Directory Server.

Root Suffix: Enter the root suffix to be used with this Directory Server.

Installation Organization: Enter the name of the installation organization as used when installing the Sun ONE Identity Server.

5. In the Sun ONE Application Server Details screen provide the following information about the Sun ONE Application Server on which the Agent is installed.

Figure 9-4 Sun ONE Application Server Details Screen

The screenshot shows a window titled "Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install Wizard". On the left is a sidebar with the Sun Microsystems logo and the text "Sun ONE Identity Server". The main area is titled "Sun ONE Application Server Details" and contains the instruction "Enter Sun ONE Application Server details." Below this are three input fields:

- Sun ONE Application Server config directory. (Value: `/var/opt/SUNWappserver7/dc`)
- Sun ONE Application Server JAVA_HOME directory (Value: `/usr/j2se`)
- Name of the Agent Realm. (Value: `AgentRealm`)

At the bottom of the window are four buttons: "<<Back", "Next>>", "Exit", and "Help".

Sun ONE Application Server config Directory: Enter the full path to the location of the config directory of the Application Server instance.

Sun ONE Application Server JAVA_HOME directory: Enter the full path to the location of the JDK installation home used by the Sun ONE Application Server.

Name of the Agent Realm: Enter the name of the Realm that will be configured as the default realm to be used by Sun ONE Application Server.

The Sun ONE Application Server `config` directory is used to modify `server.xml`, `login.conf` and `server.policy` under this directory. Default location is the following directory:

```
SIAS_Install_Dir/SUNWappserver7/domains/domain-instance/server-instance/config
```

The Sun ONE Application Server `JAVA_HOME` directory refers to the JDK installation that is used by the Sun ONE Application Server. Typically the value of this is the full path to the following directory:

```
SIAS_Install_Dir/usr/j2se
```

In case the value supplied is incorrect or is not the JDK used by the Application Server, it will result in malfunction of the Agent. To avoid this problem, ensure that the value you specify for Sun ONE Application Server `JAVA_HOME` is accurate.

-
- CAUTION**
- The installation program modifies the Application Server's `server.xml` file to include certain libraries in the Application Server `CLASSPATH`, as well as adds certain required parameters to the JVM options. Also, it adds the Agent Realm and makes it the default Realm to be used. If you specify an incorrect value for the config directory, the necessary classes and parameters will not get added, resulting in malfunction of the Agent, which can render the Sun ONE Application Server unusable. To avoid this problem, ensure that the value you specify for Sun ONE Application Server config directory is accurate.
 - The installation program modifies the Application Server's `login.conf` file to define `LoginModule` for the `jaas-context`.
 - The installation program modifies the Application Server's `server.policy` to give `ProgramaticLogin` permission to `agent-filter.jar`.
-

6. In the Agent Configuration Details screen provide the key configuration information necessary for the Agent to function correctly.

Figure 9-5 Agent Configuration Details Screen

Sun(TM) ONE Identity Server Policy Agent Version 2.0 Install

Agent Configuration Details

Enter Agent configuration values.

Agent Audit Log File:

Enable Audit log file Rotation: ☐

Host URL:

Login Attempt Limit:

Enable Not-Enforced List Cache: ☐

Number of Entries in Cache:

Cache Expiration Time in Seconds:

Enable LDAP Attribute Headers: ☐

<<Back Next>> Exit Help

Audit Log File: Enter the complete path to the log file to be used by the agent to record Audit messages.

Enable Audit log file Rotation: Select this item to enable rotation of Audit Log files.

Host URL: Enter a valid URL to be used as the base URL by the Agent to redirect users as necessary. This value may not be left blank and should have a valid FQDN of the Agent enabled Server. For example, if the Agent is installed on a WebLogic Server that may be accessed by using the URL `http://www.mycompany.com:80/`, then the Host URL should be set to `http://www.mycompany.com:80/` as well.

Login Attempt Limit: Enter the number of unsuccessful access attempts in succession after which the user will not be allowed to access the requested URL temporarily for security purposes. Specify the value 0 to disable this feature.

Enable Not-Enforced List Cache: Select this item to enable caching of Not-Enforced List evaluation results.

Number of Entries in Cache: Specify the number of entries that cache can hold at a given instance.

Cache Expiration Time: Specify the time in seconds to be used as the limit for entries that can exist in the Not-Enforced List cache.

Enable LDAP Attribute Headers: Select this item to enable the passing LDAP attributes associated with the current user as HTTP Headers.

Agent Configuration Details

- The Audit Log file is a necessary requirement for the Agent. You may provide the name of a non-existing file on the system to be used as the Audit file. In which case, the Agent creates this file and the necessary directories in its path on first use. Alternatively, you can provide the file name of an existing file, which can be used by the Agent as well. However, in either case, it is required that the specified file has write permissions for the Application Server process. This is because the Agent executes in the same process as the Sun ONE Application Server. Providing an incorrect value for this will result in the malfunction of the Agent, which can render the Application Server unusable.
- When a request is intercepted by the Agent without sufficient credentials, the Agent redirects the user to the Sun ONE Identity Server's authentication service. Along with this redirect, the Agent also passes information regarding the original request to the authentication services, which is used to redirect the user back to the original requested destination. A part of this request is the Host URL that is used to identify the web container/server which the user was originally trying to access. This Host URL can be specified by setting the Host URL value on this screen. The Host URL is also used by the Agent to provide for the default FQDN to be used in cases where required. For example, if the user types in the URL `http://mycompany/SomeApp/SomeModule` and the Host URL is set to `http://www.mycompany.com:80/`, the Agent will first redirect the user to `http://www.mycompany.com:80/SomeApp/SomeModule` before taking any other action. This will ensure that the domain specific SSO Cookie that is used by the Agent to identify the user is available. Another implication of having a Host URL is that no matter which web container/server that the user was originally trying to access, the user will be sent to the specified Host URL after successful authentication. This configuration value can also be used to override the default behavior, however, an incorrect value may lead to the application becoming

inaccessible. It is therefore recommended that this value be left to the default value chosen by the installer unless there is a specific need based on the deployment scenario, which calls for setting this value appropriately.

- When specifying the Host URL in the installation program screen, ensure that the protocol, fully qualified host name, and the port that are displayed by the installation program are valid and match the actual deployment scenario. For instance, if you intend to use the Agent to protect a Sun ONE Application Server in the SSL mode, you must ensure that the Host URL has “https” as its protocol.
- The Login Attempt Limit feature can be used to guard the hosted application from Denial-Of-Service attacks where the end user can overload the application server by repeated authentication requests. By disabling this feature, the system remains vulnerable to such attacks. Therefore this feature should not be disabled unless there is a specific requirement necessitates the disabling of this feature.
- When the Agent must process a large list of Not-Enforced pattern rules specified in the configuration, every incoming request must be evaluated against every such rule to determine if the request can be allowed without authentication or not. In scenarios where the user load is high, the time spent in evaluating these rules can add up and degrade the overall performance of the system. To avoid this problem, it is recommended that Not-Enforced List Cache be enabled.
- Enabling the Not-Enforced List Cache may result in degraded performance if the values for Number of Entries in Cache and Cache Expiration time are not set up appropriately. In case when the cache expiration time duration is more than necessary, the cache will get filled up very fast and new requests will still be evaluated against all the specified pattern rules, resulting in no improvement in performance of the system. If the Number of Entries in Cache is set very high, it may result in excessive consumption of system memory, thereby leading to degraded performance. Therefore, these values must be set up after careful deliberation of the deployment scenario and should be changed as necessary to reflect changing usage scenario of the system. It is recommended that the system be tested with various values of these two parameters in a controlled environment to identify the optimal values, and only then be deployed in production.

- The Agent maintains two caches in its memory—one for recording the URIs that were evaluated as enforced and the other for recording the URIs that were evaluated as not-enforced. The specified values of Number of Entries in Cache and Cache Expiration time are equally applicable to both of these caches. This factor must be considered when setting the values for the size and expiration time of the cache.
 - By enabling the LDAP Attribute Headers, for every incoming request, the Agent must retrieve the LDAP Attributes associated with the authenticated user and add them as Header values in the request. This feature should be used only in the case when the deployed application requires these header values for business logic implementation. Turning this feature on without an appropriate need will result in performance degradation of the system and hence can be easily avoided.
 - The parameters entered during installation can be modified later by editing the `AMAgent.properties` file. For detailed information, see the section “Agent Configuration” later in this chapter.
7. In the Summary screen review and verify the installation options that you have specified for the Agent. In case you find that certain values are in error, you may navigate back and correct them before proceeding further. Click the Next button to proceed with the installation.
 8. In the Ready to Install screen, click the Install Now button to begin the installation.
 9. The Install Progress screen displays the progress of the installation as the installer makes changes to your system. If necessary, you may interrupt this process by clicking the Stop button.

NOTE It is strongly recommended that you do not interrupt this process as this may lead to a partially installed product, which may also cause problems with uninstall, and may as well render the Application Server unusable. This process should be interrupted only in cases where it is absolutely necessary to do so.

10. In the Installation Summary window, click Details for a detailed summary of the configuration information that was processed during installation. Click Exit to end the program.

NOTE If the status of the installation is “Failed”, you must view the log file for the install by clicking on the Details button to identify which installation task failed. In this situation, you may uninstall the Agent and try again after fixing the root cause of failure during this installation.

The Agent installation program modifies the Application Server’s `server.xml` file to include certain libraries in the Application Server `CLASSPATH`, as well as adds certain required parameters to the JVM options. For the changes to take effect you must configure application server, which can be done by following either of these methods:

- To apply changes to the application server instance using the Administration interface, perform the following steps:
 - a. Access the Administration interface and click the name of the application server interface you want to reconfigure.
 - b. Click the General tab.
 - c. Click Apply Changes.

When the changes are applied, the screen displays the message “Changes applied to instance.”

- To configure an application server instance using the command-line interface, execute the following command:

```
asadmin reconfig -u admin -w password -H hostname -p admin-port
--keepmanualchanges=true server-instance
```

NOTE `asadmin` is located at `SIAS_Install_Dir/bin` directory.

Installing the Agent Using Command-Line

You must have root permissions when you run the agent installation program.

1. Run the `setup` program with the command line argument `-nodisplay`. You’ll find the program in the directory where you untarred the binaries.

```
# ./setup -nodisplay
```

2. When prompted provide the following information:

Have you read, and do you accept, all of the terms of the preceding Software License Agreement? Type Yes.

Install Sun(TM) ONE Identity Server Agent in this directory: Enter the full path to the directory in which you want to install the Policy Agent.

3. Provide the following information about the Web Server that runs Identity Server:

```
Sun(TM) ONE Identity Server Services Details.
Sun(TM) ONE Identity Server Host [nila.Eng.Siroe.COM] {"<" goes back,
"! " exits}:
Sun(TM) ONE Identity Server Port [58080] {"<" goes back, "! " exits}:
Sun(TM) ONE Identity Server Protocol [http] {"<" goes back, "! " exits}:
Sun(TM) ONE Identity Server Deployment URI [/amserver] {"<" goes back,
"! " exits}:
amAdmin Password [] {"<" goes back, "! " exits}:
Re-enter Password [] {"<" goes back, "! " exits}:
```

- Sun ONE Identity Server Host
- Sun ONE Identity Server Port
- Sun ONE Identity Server Protocol
- Sun ONE Identity Server Deployment URI
- amAdmin Password
- Re-enter Password

For details on each of these items, see “Installing the Agent Using GUI.”

4. Provide the following information about the Web Server that runs Identity Server Directory.

```
Sun(TM) ONE Identity Server Directory Details
Directory Host [nila.Eng.Siroe.COM] {"<" goes back, "!" exits}:
Directory Port [389] {"<" goes back, "!" exits}:
Root Suffix [dc=Siroe,dc=COM] {"<" goes back, "!" exits}:
Installation Organization [dc=Siroe,dc=COM] {"<" goes back, "!"
exits}:
```

- Directory Host
- Directory Port
- Root Suffix
- Installation Organization

For details on each of these items, see “Installing the Agent Using GUI.”

5. Provide the following information about the Sun ONE Application Server.

```
Sun ONE Application Server config directory.
[/var/opt/SUNWappserver7/domains/domain1/server1/config] {"<"
goes back, "!" exits}
Sun ONE Application Server JAVA_HOME directory [/usr/j2se] {"<"
goes back,"!" exits}
Name of the Agent Realm. [agentRealm] {"<" goes back, "!" exits}
```

- Sun ONE Application Server config Directory
- Sun ONE Application Server JAVA_HOME directory
- Name of the Agent Realm

For details on each of these items, see “Installing the Agent Using GUI.”

6. Provide the configuration details for the installation of Sun ONE Identity Server Agent for Sun ONE Application Server.

```
Agent Audit Log File [/var/opt/SUNWam/asAgent/audit/agent.log]
{"<" goes back, "!" exits}
Enable Audit log file Rotation [false] {"<" goes back, "!" exits}
Host URL [http://nila.Eng.Siroe.COM:80/] {"<" goes back, "!"
exits}
Login Attempt Limit [5] {"<" goes back, "!" exits}
Enable Not-Enforced List Cache [false] {"<" goes back, "!" exits}
Enable LDAP Attribute Headers [false] {"<" goes back, "!" exits}
```

- Agent Audit Log File
- Enable Audit log file Rotation
- Host URL
- Login Attempt Limit
- Enable Not-Enforced List Cache
- Enable LDAP Attribute Headers

For details on each of these items, see “Installing the Agent Using GUI.”

7. When displayed, review the summary of the installation information you have specified. Press Enter to continue.

The following text is displayed:

```
Ready to Install

1. Install Now
2. Start Over
3. Exit Installation
```

8. When prompted, **What would you like to do?** type 1 to start the installation.

The following text is displayed:

```
Installation details:
```

Product	Result	More Info
1.Sun_TM_ONE Identity Server Policy Agent	Installed	Available
2.Done		

```
Enter the number corresponding to the desired selection for more
information, or enter 2 to continue [2] {"!" exits}:
```

9. To see log information, type 1. To exit the Installation program, type 2.

Silent Installation

Silent installation provides a means for scripting the installation of Sun ONE Identity Server Policy Agent. When you perform a silent installation, you use a *StateFile*, to predefine all the answers that you would normally supply to the `setup` program interactively. This saves time and is useful when you want to install multiple instances of Sun ONE Identity Server Policy Agents using the same parameters in each instance.

Silent installation is a two-step process. First you generate a *StateFile* that contains all the parameter information the installation program needs. Then you run the silent installation program that automatically reads the parameters you've defined in the *StateFile*.

Silent Installation on Solaris

To Generate a StateFile

1. Run the installation program in the directory where the `setup` script is located. Enter the following command:

```
# ./setup -saveState StateFile
```

2. Proceed through the installation program, keeping in mind that your answers to the prompts are being recorded in the *StateFile*.

Follow the instructions given in the section “Installing the Agent Using GUI,” on page 256.

When installation is complete, the file *StateFile* is created in the same directory as *setup*.

To Run the Silent Installation

Enter the following command to run the silent installation:

```
# ./setup -nodisplay -noconsole -state StateFile
```

Silent Installation on Windows 2000

To Generate a StateFile

1. Run the installation program in the directory where the *setup* program is located. Open a DOS command prompt window and enter the following command:

```
setup.bat -saveState StateFile
```

NOTE As an alternative, you can use the following command:

```
java -classpath . agent_WINNT -saveState StateFile
```

2. Proceed through the installation program, keeping in mind that your answers to the prompts are being recorded in the *StateFile*.

Follow the instructions given in the section “Installing the Agent Using GUI,” on page 256.

When installation is complete, the file *StateFile* is created in the same directory as *setup.exe*.

To Run the Silent Installation

Type the following command:

```
setup.bat -nodisplay -noconsole -state StateFile
```

NOTE As an alternative, you can use the following command:

```
java -classpath . agent_WINNT -nodisplay -noconsole  
-state StateFile
```

Application Configuration

The Agent Realm component of Agent for Application Server 7.0 provides runtime mapping of various principals in Sun ONE Identity Server with abstract security role names used by the hosted application in order to determine if the currently authenticated user is authorized to access a particular resource or is otherwise a member of a given role. This runtime evaluation can occur only if the user is authenticated as a Sun ONE Identity Server principal by the means of Identity Server's authentication service. Without the user being authenticated appropriately, the results of such evaluations done by the Agent Realm will always be negative, resulting in access being denied to the user for the requested resource.

It is the Agent Filter component that enforces authentication for users who try to access particular application resources, thereby enabling the Agent Realm component to correctly evaluate the principal mappings as desired.

Unlike the Agent Realm component which is installed in the core of Application Server, the Agent Filter is installed in the deployed application, which must be protected by Identity Server. This is true for every application that must be protected on the Application Server using the Agent. It is also recommended that applications that are not protected using the Agent should not be deployed on the Application Server on which the Agent Realm has been installed. This is to ensure that such applications can independently enforce their own security requirements as necessary. The presence of Agent Realm will interfere with the security evaluations done by such applications resulting in their malfunction.

Installing the Agent Filter Component in an Application

The Agent Filter can be installed by simply modifying the deployment descriptor of the application that needs to be protected. The following steps outline the process to install the Agent Filter component for a given application:

1. If the application is currently deployed on the Application Server, it must be removed using the Application Server's Administration Console or by the use of Application Server's deployment tools.
2. It is recommended that you create a backup of the deployment descriptor that will be edited in order to install the Agent Filter in this application.
3. Edit the application's `web.xml` deployment descriptor. Since the Filters were introduced in Servlet Specification 2.3, the `web.xml`'s `DOCTYPE` element must be changed to reflect that the deployment descriptor is a Servlet 2.3 compliant deployment descriptor. This can be done by setting the `DOCTYPE` element as:

```
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
```

4. After the `DOCTYPE` element has been changed, add the Filter elements in the deployment descriptor. This can be done by specifying the Filter element and the Filter-mapping element in the `web.xml` deployment descriptor immediately following the description element of the `web-app` element. The following is a sample `web.xml` with the Filter and Filter-mapping elements.

```
<web-app>
  <display-name>...</display-name>
  <description>...</description>

  <filter>
    <filter-name>Agent</filter-name>
    <display-name>Agent</display-name>
    <description>Sun™ ONE Identity Server Policy Agent for Sun™ ONE
Application Server 7.0</description>
    <filter-class>com.sun.amagent.as.filter.AgentFilter</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>Agent</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  ...
  ...
</web-app>
```

5. Once the `web.xml` deployment descriptor has been modified to reflect the new `DOCTYPE` and filter elements, the Agent Filter has been added to the application.

Creating Role-to-Principal Mapping

Once the application has been configured to have the Agent Filter component in it, the Agent Filter will enforce authentication, thereby enabling the Agent Realm to successfully resolve the role-to-principal mappings. However, these mappings must first be created in order that the hosted application may use them during runtime.

To Create Role-to-Principal Mapping

- **Edit the Application Server specific deployment descriptors.**

The Application Server specific deployment descriptors should be edited to create the role-to-principal mappings. These descriptors are in the files `sun-application.xml` and `sun-ejb-jar.xml` files. Refer Application Server reference documentation to learn the details on how these descriptors may be edited to create the role-to-principal mappings. Alternatively, you can refer Appendix B for sample descriptors which create such mappings.

Application-Specific Agent Configuration

Often the deployed applications are partitioned into public and protected parts, which have varying access restrictions. In most cases, the public portions of the application are accessible to anonymous users, whereas the protected portions of the application are accessible only to the registered users. The Agent can be configured to allow this type of access by letting anonymous users access the public portions of the application without requiring that they authenticate using Identity Server's authentication service. This information is provided as a part of the Agent's configuration properties file that is present in the following location:

Agent_Install_Dir/SUNWam/asAgent/amAgent/config/AMAgent.properties

This file may be edited to provide general as well as application-specific configuration for the Agent.

The properties specified in the `AMAgent.properties` file are required for the Agent to function properly. Invalid values specified in this file can lead to malfunction of the Agent, application becoming inaccessible, or the entire system to become unusable. It is recommended that you use extreme caution when modifying the values in this file and always create backups before such modifications to ensure that you can back out your changes to restore the system to its original state.

NOTE The properties specified in the `AMAgent.properties` file are loaded during the Application Server startup time. Any changes made to this file while the Sun ONE Application Server is running will not take effect until the server has been restarted.

Providing Application Specific Not-Enforced-List

In order to allow anonymous users to access portions of the application, the `AMAgent.properties` file must include an entry for the specific application. The entry that specifies this property is called the application not-enforced list and is identified by the string:

```
com.sun.am.policy.config.filter.AppName.notEnforcedList[index]=pattern
```

This property requires that you format it correctly in order that it can be used by the Agent during runtime. The entries appearing in italics should be replaced by their appropriate values as follows:

AppName: This string should be replaced by the deployed application's context path. The context path is the first URI segment that is used to identify which application the user is trying to access. For example, if the user accesses your application by typing the URL:

```
http://myserver.mydomain.com/SomeApp/index.html or  
http://myserver.mydomain.com/SomeApp/SomeModule/doSomething.jsp
```

then, in both these cases, the ***AppName*** is `SomeApp`.

index: This is an integer value starting from 0 for every deployed application and is unique for every entry in the application not-enforced list. For example, the following are two entries of application not-enforced list for an application called with context path `SomeApp`:

```
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[0]=/public/  
*  
  
com.sun.am.policy.config.filter.SomeApp.notEnforcedList[1]=/images/  
*
```

pattern: This is a pattern string that will be matched with the incoming request to evaluate if the request should be allowed to pass without enforcing authentication or not. The pattern string could be a specific URI, for example `/public/RegistrationServlet`, or could be a generic pattern using the wild card character `'*'` that can be used for denoting 0 or more characters in the request URI; for example `/public/*` will match with any URI that begins with `/public/`.

Using this property, you could specify pattern strings and URIs that the Agent will treat as not-enforced. In other words, user requests that match these particular patterns will be allowed to pass through without enforcing authentication.

Providing Application-Specific Access Denied URI

In cases when the Login Attempt Limit is enabled (refer to section “Installing the Agent” for details on how this feature can be configured, or refer “Agent Configuration” for details on this feature), the Agent is required to block the user's access. The default behavior of the Agent in this situation is to send an HTTP Status Code 403 Forbidden. In such a situation, the web container can display its preconfigured Forbidden page or simply send the status code in which case the user's browser displays the details of the error message in its own manner. While this is the default behavior of the Agent, it can be changed to suit the needs of the application by allowing the Agent to use an application-specific URI that will be used as the access denied error page.

This can be done by setting the following property in `AMAgent.properties` file:

```
com.sun.am.policy.config.filter.AppName.accessDeniedURI=URI to use
```

This property requires that you format it correctly in order that it can be used by the Agent during runtime. The entries appearing in italics should be replaced by their appropriate values as follows:

AppName: This string should be replaced by the deployed application's context path. The context path is the first URI segment that is used to identify which application the user is trying to access. For example, if the user accesses your application by typing the URL:

```
http://myserver.mydomain.com/SomeApp/index.html or
```

```
http://myserver.mydomain.com/SomeApp/SomeModule/doSomething.jsp then,  
in both these cases, the AppName is SomeApp.
```

URI to use: is an application specific URI that the Agent will use to locate the display page for blocking the user request. This URI can be a static HTML page, or a JSP or even a Servlet. However, this URI must be a part of the application itself. In other words, this URI must begin with */AppName/rest of the URI*.

Special Case: Default Web Application

A default web application in Application Server is accessible without providing any context path in the request URI. For example, the following URL is that of a default web application:

`http://myserver.mydomain.com/index.html`

Clearly, this URL does not have an associated context path.

For such applications, the Agent provides a convenient means of identifying that an entry is specific to the default web application. This is done in two steps as follows:

1. The following property is set to a name that represents the default web application:

```
com.sun.am.policy.config.filter.defaultWebAppName=DefaultWebApp
```

2. This name is then used to specify the application not-enforced list as well as the application's access denied URI as follows:

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList
[0]=/index.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]
=/about.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.accessDeniedURI=/U
RLAccessDenied.html
```

Using this scheme, the default web application that does not have a context path associated with it, may be configured just like any other application that has a context path. The same rules apply to the default web application for specifying the not-enforced list entries and access-denied URI as are applicable for the rest of the application. However, the only difference is that the access-denied URI of the default web application as well as the not-enforced list entries do not begin with the `/DefaultWebApp/` path segment since such a path segment does not exist on the application server in reality. It is only provided as a convenience to specify the properties associated with the default web application and not their values.

Global Agent Configuration

The `AMAgent.properties` file provides a way to specify a global not-enforced list, which will be applicable to all the deployed applications on the server. This list is specified by using the following property:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=pattern
```

where *pattern* is either an exact URI or a pattern specified by using the wild card character `'*'`, which can be substituted for zero or more characters in the request URI.

Not-Enforced List Usage Considerations

Although the use of Not-Enforced List can be extremely helpful in partitioning your application for public and protected domains, it can also lead to undesirable effects if not used appropriately.

For example, if a request URI that represents a Servlet is matched by some not-enforced list pattern, then the Agent Filter will not enforce authentication for users who try to access that particular Servlet. However, consider the case where this Servlet access an Enterprise JavaBeans component that is protected by the Agent using role-to-principal mapping. In such a case, since the user is not authenticated, the access to the protected component will result in a an security violation exception being generated by the application server.

Therefore, before an entry is added to the not-enforced list, it must be ensured that it does in any way covers a resource that may be protected or may try to access a protected resource.

Another interesting aspect of the use of non-enforced list are the images. Typically, in a web page there are many images for various purposes like buttons, place holders, banners and logos. Every time the user accesses this page, the browser issues a request to the application server to get the images contained in this page. Each of such requests are treated as individual requests coming from the client and goes through the same evaluation mechanism for authentication and not-enforced list check as does any other request. This results in one client generating multiple calls to the server for displaying of a single page. Considering the overhead involved in enforcing authentication for every such request, it can impact the overall performance of the system. A solution to this problem is to have a global not-enforced list entry or entries that match all images. For example:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=*.gif  
com.sun.am.policy.config.filter.global.notEnforcedList[1]=/images/*
```

This indicates that any request URI that ends with `.gif` will be not enforced and nor will be any URI that begins with `/images/`. In heavy user load situations, this can significantly increase the performance of the system.

Agent Configuration

The file `AMAgent.properties` located at:

`Agent_Install_Dir/SUNWam/asAgent/amAgent/config` directory provides the core configuration needed by the Sun ONE Identity Server Policy Agent for Sun ONE Application Server 7.0 to function correctly. This property file provides many configuration settings that can be modified in order to customize the Agent's operation for your deployment scenario. This section provides a brief explanation of all the properties that are listed in the properties file.

Before proceeding, it is important to note that this file and the information within it are critical for the operation of the Agent. It is strongly recommended that you always create backup of this file before modifying it. Also it is strongly recommended that you do not modify this file unless you it is absolutely necessary. Please note that invalid data entries present in this file can lead to the malfunction of the Agent, the malfunction of the deployed applications, and could render the entire system unusable.

The settings provided in this file can be classified as follows:

- **Common Configuration:** Settings in this section are general settings that affect the behavior of the Agent as a whole.
- **Audit Configuration:** These settings are exclusively used to configure the Audit Engine used by the Agent.
- **Global Filter Configuration:** These settings are used to configure the Agent Filter component.
- **Application Filter Configuration:** These settings are used to configure the Agent Filter for a particular application.
- **Debug Engine Configuration:** These settings are used to configure the Debug Engine to generate diagnostic information.

Common Configuration

Organization Name

Key: `com.sun.am.policy.config.org`

Description: This property specifies the organization name to be used when searching for principals in Identity Server.

Valid Values: A string that represents the organization name in Sun ONE Identity Server

NOTE	This property is set during Agent installation and need not be changed unless absolutely necessary.
-------------	---

Example: `com.sun.am.policy.config.org=sun.com`

Root Suffix

Key: `com.sun.am.policy.config.rootsuffix`

Description: This property specifies the root suffix to be used when searching for principals in Sun ONE Identity Server

Valid Values: A string that represents the root suffix in Sun ONE Identity Server

NOTE	This property is set during Agent installation and need not be changed unless absolutely necessary.
-------------	---

Example: `com.sun.am.policy.config.rootsuffix=o=isp`

People Container Level

Key: `com.sun.am.policy.config.realm.peopleContainerLevel`

Description: This property specifies the people container level to be used when searching for principals in Sun ONE Identity Server.

Valid Values: Non-zero unsigned integer representing the People Container Level in Identity Server, which may be used when searching for principals.

NOTE	This property is set during Agent installation and need not be changed unless absolutely necessary.
-------------	---

Example: `com.sun.am.policy.config.realm.peopleContainerLevel=1`

Audit Configuration

Language Code

Key: `com.sun.am.policy.config.audit.localeLanguageCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeLanguageCode` must be a valid ISO Language Code. Default value of this property is `en`

NOTE For more information, refer ISO 639 specification:
<http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt>

Example: `com.sun.am.policy.config.audit.localeLanguageCode=en`

Country Code

Key: `com.sun.am.policy.config.audit.localeCountryCode`

Description: This property specifies the Locale for Audit log messages.

Valid Values: The `localeCountryCode` must be a valid ISO Country Code. Default value of this property is `US`.

NOTE For more information, refer ISO 3166 specification:
http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html

Example: `com.sun.am.policy.config.audit.localeCountryCode=US`

Audit Log File

Key: `com.sun.am.policy.config.audit.logfile.name`

Description: This property specifies the Audit log file to be used for recording Audit messages.

Valid Values: String representing the complete path name of the file to be used by Agent to record Audit messages.

NOTE Be sure the Application Server process has sufficient permissions to write to this file. Invalid value specified for this property may result in the failure of the system to start up correctly.

Example:
`com.sun.am.policy.config.audit.logfile.name=/var/opt/SUNWam/asAgent/audit/agent.log`

Audit Log File Rotation Flag

Key: `com.sun.am.policy.config.audit.logfile.rotate`

Description: This property specifies if the Audit log file should be rotated by the Agent.

Valid Values: `true/false`

NOTE Default value of this property is `false` and should be changed as necessary.

Example: `com.sun.am.policy.config.audit.logfile.rotate=false`

Audit Log File Rotation Size

Key: `com.sun.am.policy.config.audit.logfile.rotate.size`

Description: This property specifies the approximate size of the Audit log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated.

NOTE Audit Log file rotation size is effective only when rotation flag is `true`.

Default value of this property is 52428800 bytes (~ 50 MB) and should be changed as necessary.

Example: `com.sun.am.policy.config.audit.logfile.rotate.size=52428800`

Realm Configuration

These settings are used to configure the Agent Realm component.

SSO Cache Cleanup Size

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupSize`

Description: This property specifies the maximum number of entries in the SSO Cache maintained by the Agent Realm after which a cleanup will be initiated.

Valid Values: Any positive integer value indicating the number of entries in the Agent Realm's SSO Cache which when exceeded will initiate a clean up operation to ensure minimal and appropriate use of the system's memory.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance and/or result in the loss of availability of SSO token in the system.
 - The value for optimal system performance will depend on the type of application deployed, the number of active user sessions that the application is subject to during peak periods, and the average user session time value.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 1000.
-

Example: `com.sun.am.policy.config.realm.ssoCacheCleanupSize = 1000`

SSO Cache Cleanup Lock Time

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime`

Description: This property specifies the amount of time in seconds that the Agent Realm will wait before initiating the next cleanup process for the SSO cache if the last cleanup process did not result in freeing any memory.

Valid Values: Any positive integer value indicating the number of seconds that the cleanup process will not be restarted if the last cleanup process did not result in freeing sufficient memory.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
 - The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 1200.
-

Example: `com.sun.am.policy.config.realm.ssoCacheCleanupLockTime = 1200`

SSO Cache Cleanup Bound Size

Key: `com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize`

Description: This property specifies the maximum number of entries in the Agent Realm SSO Cache that will be inspected during cleanup process. This value when set appropriately will result in overall improvement of the response time of the system when it undergoes a cache cleanup operation.

Valid Values: Any positive integer value indicating the number of entries the cleanup process will inspect in the SSO cache during the cleanup operation. This value can be set independently with respect to the value of the property `com.sun.am.policy.config.realm.ssoCacheCleanupSize`.

NOTE

- Values that are valid but not suited for the deployment scenario may result in degradation of system performance.
 - The value for optimal system performance will depend on the type of application deployed and the typical session time for an average user using this application.
 - To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.
 - This property defaults to a value of 50.
-

Example:

`com.sun.am.policy.config.realm.ssoCacheCleanupBoundSize = 50`

Global Filter Configuration

SSO Token Name

Key: `com.sun.am.policy.config.filter.ssoTokenName`

Description: This property specifies the name of the Cookie that represents SSO Token.

Valid Values: A string that represents the name of SSO Token Cookie issued by Sun ONE Identity Server authentication service.

NOTE	This property is set during Agent installation and need not be changed unless absolutely necessary.
-------------	---

Example:

```
com.sun.am.policy.config.filter.ssoTokenName=iPlanetDirectoryPro
```

FQDN Map

Key:

```
com.sun.am.policy.config.filter[invalid-name]
```

Description: The FQDN Map is a simple map that enables the Agent to take corrective action in the case where the users may have typed in an incorrect URL such as by specifying partial hostname or using an IP address to access protected resources.

Valid Values: Valid values must comply with the syntax of this property which represent invalid FQDN values mapped to their corresponding valid counterparts.

The format for specifying this property is as follows:

```
com.sun.am.policy.config.filter.fqdnMap[invalid-name]=valid-name
```

Where *invalid-name* is a possible invalid FQDN host name that may be used by the user, and the *valid-name* is the FQDN host name the filter will redirect the user to.

NOTE

- Ensure that there are no overlapping values for the same invalid FQDN name. Failing to do so may lead to the application becoming inaccessible.
- Invalid value for this property can result in the application becoming inaccessible.
- This property can be used for creating a mapping for more than one host name. This may be the case when the applications hosted on this server are accessible by more than one host name. However, the use of this feature must be done with caution as it can lead to the applications becoming inaccessible.
- The Agent gives precedence to the entries defined in this property over the `com.sun.am.policy.config.filter.hostURL` property.
- This property may be used to configure the Agent to not take corrective action for certain hostname URLs. For example, if it is required that no corrective action such as a redirect be used for users who access the application resources by using the raw IP address, you can implement this by specifying a map entry such as:

`com.sun.am.policy.config.filter[IP]=IP`
- Any number of such properties may be specified as long as they are valid properties conforming to the above stated requirements.

Example:

```
com.sun.am.policy.config.filter[myserver]=myserver.mydomain.com
com.sun.am.policy.config.filter[myserver.mydomain]=myserver.mydomain.com
com.sun.am.policy.config.filter[IP]=myserver.mydomain.com
com.sun.am.policy.config.filter[invalid-name]=valid-name
```

Login URL

Key: `com.sun.am.policy.config.filter.loginURL`

Description: This property specifies the login URL to be used by the Agent to redirect incoming users without sufficient credentials to the Sun ONE Identity Server authentication service.

Valid Values: A string that represents the complete URL to be used as the redirect URL in order to send users without sufficient credentials to Sun ONE Identity Server authentication service.

NOTE This property is set during Agent installation and need not be changed unless absolutely necessary.

Example:

```
com.sun.am.policy.config.filter.loginURL=http://myserver.mydomain.com:58080/amserver/UI/login
```

Host URL

Key: `com.sun.am.policy.config.filter.hostURL`

Description: This property specifies the host URL to be used by the Agent to reconstruct the request issued by the browser. This value is used by Sun ONE Identity Server Authentication service to redirect the user back to the original request destination after successful authentication.

Valid Values: A string value that represents the host URL that the user is expected to access in order that the Agent can intercept. This value must match the format of this property. The format of this property value is as follows:

protocol: // *hostname*. *optional-sub-domain*. *domain*: *port*

protocol can be http or https.

hostname.optional-sub-domain.domain is the fully qualified host name that the user is expected to access in order that the Agent may intercept.

port is the port number on which the receiving web server is listening.

If left unspecified, the Agent will try to reconstruct the host URL from the request.

Example:

```
com.sun.am.policy.config.filter.hostURL=http://www.ipplanet.com:80
```

Goto Parameter

Key: `com.sun.am.policy.config.filter.gotoParameter`

Description: This property specifies the goto parameter name to be used by the Agent when redirecting the user to the appropriate authentication service. The value of this parameter is used by the authentication service to redirect the user to the original requested destination.

Valid Values: A string value that represents the goto parameter name as expected by the authentication service.

Example: `com.sun.am.policy.config.filter.gotoParameter=goto`

Login Attempt Limit

Key: `com.sun.am.policy.config.filter.loginAttemptLimit`

Description: This property specifies the number of login attempts that a user can make without success using a single browser session, which will trigger the blocking of the user request.

Valid Values: Unsigned integer value, including 0, which indicates the number of unsuccessful login attempts that are allowed for any user trying to gain access to protected resources.

NOTE This option can be disabled by setting the value to 0. The default value of this property is 5.

Example: `com.sun.am.policy.config.filter.loginAttemptLimit=5`

Login Counter Cookie Name

Key: `com.sun.am.policy.config.filter.loginCounterCookieName`

Description: This property specifies the name of the cookie that will be used to track the number of unsuccessful login attempts made by the user.

Valid Values: A string that represents the name of the cookie to be issued by Agent in order to track the number of unsuccessful login attempts made by the user.

NOTE This property is set during Agent installation and need not be changed unless absolutely necessary.

Example:

`com.sun.am.policy.config.filter.loginCounterCookieName=iPlanetLoginAttemptID`

Not-Enforced-List Cache Enable Flag

Key: `com.sun.am.policy.config.filter.notEnforcedList.cache`

Description: This property specifies if the requests URIs that are evaluated as enforced or not-enforced may be cached to increase performance of the system.

Valid Values: `true/false`

NOTE The default value of this property is `true`.

Example: `com.sun.am.policy.config.filter.notEnforcedList.cache=true`

Not-Enforced-List Cache Size

Key: `com.sun.am.policy.config.filter.notEnforcedList.cacheSize`

Description: This property specifies the number of entries that will be kept in the cache of not-enforced URIs and enforced URIs by the Agent.

Valid Values: Non-zero unsigned integer indicating the number of enforced as well as not enforced request URIs to be cached during runtime.

NOTE Values that are valid but not suited for the deployment scenario may result in degradation of system performance.

The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries and a host of other deployment specific factors.

To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.

Example:

`com.sun.am.policy.config.filter.notEnforcedList.cacheSize=1000`

Not-Enforced-List Cache Expiration Time

Key: `com.sun.am.policy.config.filter.notEnforcedList.cacheTime`

Description: This property specifies the amount of time in seconds that will be used to evaluate if a cached entry can be removed from the cache to free up resources for new cache entries.

Valid Values: Non-zero unsigned integer indicating the time in seconds that will be used as the cache expiration time for entries in the cache during cleanup operation.

Values that are valid but not suited for the deployment scenario may result in degradation of system performance.

The value for optimal system performance will depend on the type of application deployed, the number of possible request URIs in the deployed application, the user load on the system, the expiration time set for cache entries and a host of other deployment specific factors.

To determine the most optimal value of this property, the application should be load tested in a controlled test environment before being deployed for production.

Example:

```
com.sun.am.policy.config.filter.notEnforcedList.cacheTime=60
```

LDAP Attribute Header Enable Flag

Key: `com.sun.am.policy.config.filter.enableLDAPAttributeHeaders`

Description: This property specifies if the Agent should populate the `HttpServletRequest` with LDAP Attributes associated with the currently authenticated user.

Valid Values: `true/false`

The default value of this property is `false` and should be changed as necessary.

Example:

```
com.sun.am.policy.config.filter.enableLDAPAttributeHeaders=true
```

LDAP Attribute Header Map

Key: `com.sun.am.policy.config.filter.ldapAttribute[attr-name]`

Description: This property specifies the LDAP Attributes to be populated under specific header names for the currently authenticated user.

Valid Values: Valid values must comply with the syntax of this property. The specified LDAP Attribute should be a valid attribute. The specified HTTP Header name should conform to HTTP Header name conventions.

The format for specifying this property is as follows:

`com.sun.am.policy.config.filter.ldapAttribute[attr-name]=header-name`

attr-name is the name of the LDAP Attribute to be looked up for the authenticated user, and *header-name* is the name of the Header that will be used to store this value.

NOTE

- Be sure that the specified Header Names do not conflict with the existing Header names.
 - Any number of such properties may be specified as long as they are valid properties conforming to the above stated requirements.
-

Example:

`com.sun.am.policy.config.filter.ldapAttribute[cn]=CUSTOM-Common-Name`

`com.sun.am.policy.config.filter.ldapAttribute[ou]=CUSTOM-Organization-Unit`

`com.sun.am.policy.config.filter.ldapAttribute[o]=CUSTOM-Organization`

`com.sun.am.policy.config.filter.ldapAttribute[c]=CUSTOM-Country`

`com.sun.am.policy.config.filter.ldapAttribute[mail]=CUSTOM-Email`

`com.sun.am.policy.config.filter.ldapAttribute[employeenumber]=CUSTOM-Employee-Number`

LDAP Date Header Attribute Format String

Key: `com.sun.am.policy.config.filter.ldapAttributeDateHeaderFormat`

Description: This property specifies the format of Date/Time value to be expected as a result of an attribute lookup. This is required when using the specialized get methods of the

`javax.servlet.http.HttpServletRequest` interface that return Date values for headers.

Valid Values: Valid `java.text.SimpleDateFormat` Time Format Syntax string. For more information, see: documentation at

<http://java.sun.com/j2se/1.4/docs/api/java/text/SimpleDateFormat.html>

NOTE The default value of this property is set to `EEE, d MMM yyyy hh:mm:ss z` and should be changed as necessary.

Invalid value of this property may result in runtime exceptions in the application.

Example:

```
com.sun.am.policy.config.filter.ldapAttributeDateHeaderFormat=EEE,
d MMM yyyy hh:mm:ss z
```

SSO Token URL Decode Flag

Key: `com.sun.am.policy.config.filter.urlDecodeSSOToken`

Description: This property indicates if the SSO Token needs to be URL Decoded by the Agent before it may be used.

Valid Values: `true/false`

NOTE The default value of this property is set to `true`

Example: `com.sun.am.policy.config.filter.urlDecodeSSOToken=true`

Default Web Application Name

Key: `com.sun.am.policy.config.filter.defaultWebAppName`

Description: This property specifies a name for the Default Web Application deployed on the application server.

Valid Values: A string consisting of lower case and or upper case letters that can be used as the name for default web application.

NOTE

- The default value of this property is `DefaultWebApp`
- This property is necessary if the protected application is deployed as the default web application.

Example:

```
com.sun.am.policy.config.filter.defaultWebAppName=DefaultWebApp
```

Global Not-Enforced List

Key: `com.sun.am.policy.config.filter.global.notEnforcedList[index]`

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters.

The syntax of this property is as follows:

```
com.sun.am.policy.config.filter.global.notEnforcedList[index]=pattern
```

index is an integer that starts from 0 and increments for every entry in this property list.

pattern is a string that represents request URIs that are not enforced by Agent.

NOTE

- The *pattern* string may consist of wild card character '*', which may match zero or more characters.
 - The *index* must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries.
 - No value for this property indicates empty global not enforced list.
-

Example:

```
com.sun.am.policy.config.filter.global.notEnforcedList[0]=*.gif
com.sun.am.policy.config.filter.global.notEnforcedList[1]=/public/*
com.sun.am.policy.config.filter.global.notEnforcedList[2]=/images/*
```

Application Filter Configuration

Access Denied URI

Key: `com.sun.am.policy.config.filter.AppName.accessDeniedURI`

Description: This property specifies the application specific Access Denied URI for the protected application.

Valid Values: The URI within the deployed application that must be used as the access denied URI to block in coming requests when necessary.

This property is specific to the protected application. Therefore if there are more than one protected applications deployed on the system, there should be one property for each such application.

Note: This property must specify a URI that is within the application. Failing to do so can result in runtime internal server errors.

Note: The format for specifying this property is as follows:

```
com.sun.am.policy.config.filter.AppName.accessDeniedURI=URI
```

AppName is the context path name for the deployed application and URI is the URI to be used.

-
- | | |
|-------------|--|
| NOTE | <ul style="list-style-type: none"> • The difference between <i>AppName</i> and context path of the application is that the context path has a leading / character. • If the protected application is the default web application, the <i>AppName</i> should be set to the same string as specified in for the value of the property Default Web Application Name. • If the property is not specified for a given application, the Agent uses HTTP Status Code 403 (Forbidden) to indicate a blocked access. |
|-------------|--|
-

Example:

```
com.sun.am.policy.config.filter.Portal.accessDeniedURI=/Portal/AccessDenied.html
```

```
com.sun.am.policy.config.filter.BankApp.accessDeniedURI=/BankApp/Block.jsp
```

```
com.sun.am.policy.config.filter.DefaultWebApp.accessDeniedURI=/URLAccessDenied.htm
```

Application Not-Enforced-List

Key: `com.sun.am.policy.config.filter.AppName.notEnforcedList[index]`

Description: This property specifies a list of patterns that can be used to evaluate if the requested URI does not require the protection enforced by the Agent for a particular application.

Valid Values: Valid values must comply with the syntax of this property. The valid values can be exact URIs or patterns consisting of wild-card character '*' to indicate zero or more characters.

The syntax of this property is as follows:

```
com.sun.am.policy.config.filter.AppName.notEnforcedList[index]=pattern
```

AppName is the context path name without the leading '/' character for the deployed application.

index is an integer that starts from 0 and increments for every specified property for the particular application.

pattern is a string that represents the URIs that are not enforced by the Agent.

NOTE

- The *pattern* string may consist of wild card character '*', which may match zero or more characters.
 - The *index* must start from zero for the first entry and continue till the last in a sequence. Missing index values in this list will result in partial or complete loss of list entries. The index value will be independent for different *AppName* specified in this property list.
 - In case the protected application is the default web application, the *AppName* should be set to the same string as specified in for the value of the property Default Web Application Name.
 - No value for this property indicates empty not-enforced list.
-

Example:

```
com.sun.am.policy.config.filter.Portal.notEnforcedList[0]=/Portal/GuestPages/*
```

```
com.sun.am.policy.config.filter.Portal.notEnforcedList[1]=/Portal/Registration/*
```

```
com.sun.am.policy.config.filter.Portal.notEnforcedList[2]=/Portal/WebServices/PollServlet
```

```
com.sun.am.policy.config.filter.BankApp.notEnforcedList[0]=/BankApp/ModuleGuestTour/*
```

```
com.sun.am.policy.config.filter.BankApp.notEnforcedList[1]=/BankApp/index.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[0]=/index.html
```

```
com.sun.am.policy.config.filter.DefaultWebApp.notEnforcedList[1]=/about.html
```

Debug Engine Configuration

Debug Level

Key: `com.sun.am.policy.config.debug.level`

Description: This property specifies the amount of debug messages that will be emitted by the Agent's Debug Engine.

Valid Values: Any of 0, 1, 3, 7, 15, and 31. These values indicate the following:

- 0 : No debugging
- 1 : Only Error messages
- 3 : Error and Warning messages
- 7 : Error, Warning and Brief Informational messages
- 15: Error, Warning and Verbose Informational messages
- 31: Error, Warning and Very Verbose Informational messages

-
- | | |
|-------------|--|
| NOTE | <ul style="list-style-type: none"> • For better performance of the system, this property should be set to a value 0. Values other than that will affect the system performance depending upon the amount of information the Debug Engine has to emit. • Values other than the ones specified in the Valid Values list above will lead to invalid configuration of the Debug Engine, thereby affecting the volume of messages that will be emitted. |
|-------------|--|
-

Example: `com.sun.am.policy.config.debug.level=7`

Debug Log File

Key: `com.sun.am.policy.config.debug.logfile.name`

Description: This property specifies the Debug log file to be used for recording Debug messages.

Valid Values: String representing the complete path name of the file to be used by Agent to record Debug messages.

-
- | | |
|-------------|---|
| NOTE | <ul style="list-style-type: none"> • Be sure that the Application Server process has sufficient permissions to be able to write to this file. • Invalid or empty value of this property will lead to Debug messages not being logged in the log file. |
|-------------|---|
-

Example:

```
com.sun.am.policy.config.debug.logfile.name=/debug/agent_debug.log
```

Debug Log File Rotation Flag

Key: `com.sun.am.policy.config.debug.logfile.rotate`

Description: This property specifies if the Debug log file should be rotated by the Agent.

Valid Values: `true/false`

-
- | | |
|-------------|---|
| NOTE | Default value of this property is false and should be changed as necessary. |
|-------------|---|
-

Example: `com.sun.am.policy.config.debug.logfile.rotate=false`

Debug Log File Rotation Size

Key: `com.sun.am.policy.config.debug.logfile.rotate.size`

Description: This property specifies the approximate size of the Debug log file in bytes, which should be used to evaluate when the log file needs to be rotated.

Valid Values: Non-zero unsigned integer indicating the size in bytes to be used to evaluate when the log file needs to be rotated.

-
- | | |
|-------------|---|
| NOTE | <ul style="list-style-type: none"> • Default value of this property is 52428800 bytes (~ 50 MB) and should be changed as necessary. • This property is not used if the Debug Log File Rotation Flag is set to false |
|-------------|---|
-

Example: `com.sun.am.policy.config.debug.logfile.rotate.size=52428800`

Debug Time/Date Format String

Key: `com.sun.am.policy.config.debug.date.format`

Description: This property specifies the format of time stamp that is used to mark the exact time when the Debug message was recorded.

Valid Values: Valid `java.text.SimpleDateFormat` Time Format Syntax **string**. For more information, see URL:

<http://java.sun.com/j2se/1.4/docs/api/java/text/SimpleDateFormat.html>

NOTE The default value of this property is set to:

MMM d, yyyy h:mm:ss a z 'Agent' and should be changed as necessary.

Invalid value of this property will result in loss of time stamp data with debug messages.

Example:

`com.sun.am.policy.config.debug.date.format=[yyyy/MM/dd HH:mm:ss zzz]`

Debug Print STDOUT Flag

Key: `com.sun.am.policy.config.debug.print.stdout`

Description: This property specifies if the Debug Engine should print the debug messages on Standard Output stream.

Valid Values: `true/false`

NOTE • The default value of this property is `true` and should be changed as necessary. When set to `true`, the Debug Engine prints all debug messages on the Standard output stream. This results in the debug messages being displayed on the console window where the Application Server startup script was executed from.

• This property has no affect on the ability of Debug Engine to write to debug log files.

Example: `com.sun.am.policy.config.debug.print.stdout=true`

Using Agent and Sun ONE Identity Server SDK APIs

You can use the Sun ONE Identity Server SDK APIs to create security and identity aware applications. Such applications can perform custom security and identity related tasks such as application level policy enforcement by exploiting the rich security and policy infrastructure offered by the Sun ONE Identity Server. When you install the Policy Agent for Sun ONE Application Server, the Sun ONE Identity Server SDK becomes available for your application to use.

While the availability of the SDK in itself is sufficient for the developers of the application to make it security aware, the Policy Agent further facilitates this by ensuring that the Single Sign-On (SSO) Token is available for the logged on user who at any given point in time may be using the deployed system.

When the logged on user accesses a protected resource, the Agent Filter ensures that the user be appropriately authenticated and that the corresponding Principal be available throughout the system. The Principal instance may be accessed by the J2EE programmatic security calls such as

`HttpServletRequest.getUserPrincipal()` and

`EJBContext.getCallerPrincipal()`. The Principal instance returned by these methods represent the user as authenticated by the Sun ONE Identity Server's authentication service. This Principal instance can then be used to access the user's SSO Token from anywhere within the application in the following two simple steps:

1. Downcast the Principal to the class of type:

```
com.sun.am.policy.as.realm.AgentUser
```

For example:

```
....
import java.security.Principal;
import com.sun.am.policy.as.realm.AgentUser;
....
Principal principal = getEJBContext().getCallerPrincipal();
AgentUser user = null;
if (principal instanceof AgentUser) {
    user = (AgentUser) principal;
}
...
```

2. Use the `AgentUser` API to retrieve the SSO Token associated with this principal. For example:

```
...
String ssoTokenId = null;
if (user != null) {
    ssoTokenId = user.getSSTokenID();
}
...
```

Once the SSO Token string is acquired, it can be used for subsequent calls into the Identity Server SDK APIs.

NOTE The Agent uses these configuration settings which ensure the availability of the SSO token associated with the user in the Agent Realm. If the values for these settings is not appropriate for your deployment scenario, it may result in the degradation of the overall system performance. See “Agent Configuration.”

Uninstalling the Agent

When you install the Sun ONE Identity Server Policy Agent for Sun ONE Application Server software, an uninstallation program is created in the installation directory. Using this uninstallation program the Agent can be removed completely from your system. While the uninstallation program deletes all the installed files from your system, certain files such as audit log messages are not deleted. You can delete them manually.

The uninstallation program for Sun ONE Identity Server Policy Agent for Application Server should be launched according to the following steps for Solaris or Windows platform as applicable.

Launching the Uninstallation Program on Solaris

The uninstallation program for Solaris platform may be launched by executing the generated uninstall script located in the installation directory. The following steps provide details on how to achieve this:

1. Login as root.
2. Stop the Application Server instance using the following command:

```
/SIAS_Install_Dir/appserv/domains/domain-instance/server-instance/bin/stopse  
rv
```

3. Set your `JAVA_HOME` environment variable to JDK version 1.4.0 or higher. If your system does not have JDK of required version, use the JDK supplied with Sun ONE Application Server. This JDK is located under:

```
SIAS_Install_Dir/usr/j2se
```

4. The uninstallation program provides two types of interfaces—a graphical user interface (GUI) and a command-line. In most cases, the GUI installer can be used for uninstalling the Agent. However, in cases when you are uninstalling the Agent over a telnet session on a remote server and do not have windowing capabilities, then the GUI uninstallation program cannot be used. In such a case it is recommended that you use the command line uninstallation program for uninstalling the Agent. This can be launched by executing the `uninstall` script and passing in a command line argument `-nodisplay` as follows:

```
#./uninstall_asagent -nodisplay
```

However, if you choose to use the GUI uninstallation program, then it is required that you set your `DISPLAY` environment variable to ensure that the GUI uninstallation program window appears on the correct console.

NOTE	If you choose to use the command-line uninstallation program using the <code>-nodisplay</code> option, you may skip the following step and proceed directly to the next section, which details out the uninstallation procedure.
-------------	--

5. Launch the GUI uninstallation program by invoking the `uninstall` script as follows:

```
# ./uninstall_asagent
```

The uninstallation program requires that you setup your `JAVA_HOME` variable correctly as pointed out in the Step 3. However, in case you have incorrectly set the `JAVA_HOME` variable, the `uninstall` script will prompt you for supplying the correct `JAVA_HOME` value:

```
Enter JAVA_HOME location (Enter "." to abort):
```

Type the full path to the JDK installation directory for launching the installation program. Otherwise, enter a period (.) to abort the uninstallation.

In order that the GUI uninstallation program be displayed on your console, the `DISPLAY` environment variable of your shell must be set correctly. In case your `DISPLAY` environment variable is not set at the time of invoking the uninstall script, the uninstallation program will prompt you for the `DISPLAY` environment variable value as follows:

Please enter the value of `DISPLAY` variable (Enter "." to abort):

Provide the `DISPLAY` value to the installer by typing in the exact value at the above prompt. Otherwise, enter a period (.) to abort the installation.

Launching the Uninstallation Program on Windows 2000 Server

The uninstallation program for the Windows 2000 Server platform may be launched by executing the generated uninstallation script located in the installation directory.

1. You must have administrative privileges when you run the uninstallation program. If you do not have administrative privileges, either login as "Administrator" or request such privileges to be granted to your account by the system administrator of the machine or domain as applicable.
2. Stop the Application Server instance using the following command:

```
SIAS_Install_Dir\appserv\domains\domain-instance\server-instance\bin\stop
serv
```

3. Go to the directory where Agent is installed.
4. The uninstall script `uninstall_asagent.bat` is located in this directory. In order to use the `uninstall_asagent.bat` script to launch the uninstallation program, you must have JDK version 1.4.0 or higher. This can be verified by typing the following command in a command prompt window:

```
C:\> java -version

java version "1.4.0_02"

Java(TM) 2 Runtime Environment, Standard Edition (build
1.4.0_02-b03)

Java HotSpot(TM) Client VM (build 1.4.0_02-b03, mixed mode)
```

If you do not have JDK of required version in your system path, you can use the JDK supplied with Application Server located at:

```
SIAS_Install_Dir\jdk
```

The `uninstall_asagent.bat` may be executed by typing the file name at the command prompt window in a directory where it is present, or by double clicking the file in Windows Explorer. For example:

```
C:\Sun\uninstall_asagent.bat
```

The uninstallation program provides two types of interfaces—a graphical user interface (GUI) and a command line interface. By invoking the `uninstall_asagent.bat` file from a command prompt window as shown above or by double clicking it in Windows Explorer, the uninstallation program is launched in the GUI mode. However, in a case where it is required that you use the command line based uninstallation program for uninstalling the Agent, the uninstallation program may be launched by executing the `uninstall_asagent.bat` file and passing in a command line argument `-nodisplay` as follows:

```
Agent_Install_Dir\uninstall_asagent.bat -nodisplay
```

NOTE

- If the required JDK is configured to be in the system path, the uninstallation program can be launched from the Control Panel Add/Remove programs control. In the list of the installed programs on your system, select Sun ONE Identity Server Policy Agent for Application Server and click the Change/Remove button. If your system path is not configured with an appropriate JDK version, the uninstall can fail.
 - When using the Control Panel Add/Remove programs to launch the uninstallation program for Agent, the uninstallation program will be launched in GUI mode only. To launch the uninstallation program in command line mode, use the provided uninstall script as mentioned previously.
-

Uninstalling the Agent Using GUI

The uninstallation program begins with a Welcome screen.

5. Click Next to step through the uninstallation screens, answering the questions.
6. In the Ready to Uninstall screen, review the uninstallation information. If you need to make changes, click Back. Otherwise, click Uninstall Now.

The Uninstall Progress screen displays the progress of uninstall process.

7. In the Uninstallation Summary window, click Details for a detailed summary of the configuration information that was processed during uninstallation. Click Exit to end the program.

NOTE

If the status of the uninstall is “Failed”, you must view the log file for the uninstall by clicking on the Details button to identify which uninstall task failed. In certain situations these failures can be recovered from and the system can be restored to its original state. Refer to the next section for the detail of how to restore the system to its original state.

Uninstalling the Agent Using Command-Line

You must have root permissions when you run the agent installation program.

1. Run the uninstall program with the command line argument `-nodisplay`. You'll find the program in the directory where you have installed the agent.

```
# ./uninstall_asagent -nodisplay
```

2. The following text is displayed:

```
Sun(TM) ONE Identity Server Policy Agent for Sun(TM) ONE
Application Server 7.0

Ready to Uninstall

1. Uninstall Now
2. Start Over
3. Exit
Choose one option from above [1] {"<" goes back, "!" exits}
```

3. Enter 1 to uninstall.

The following text is displayed:

```
Uninstallation Summary

Installation summary      Summary Result More Details
1. Sun_TM_ONE Identity Server Policy  Full          Enter 1 to view the log

Done
```


4. To see log information, enter 1. To exit the Uninstallation program, press Enter.

Configuration Tasks Performed by Installer

This appendix explains how to configure the following application servers to recover from installation failure:

- WebLogic 6.1 SP2
- WebSphere 4.0.4 AE
- Sun ONE Application Server 7.0

WebLogic 6.1 SP2

The Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 Server Installer performs certain configuration tasks. Depending on your system configuration and various other factors, these tasks can occasionally fail resulting in a unusable installation. Fortunately, such failures in most cases are recoverable by manually performing these tasks. The following sections describe how to configure Sun ONE Identity Server Policy Agent for WebLogic 6.1 SP2 Installer.

WebLogic Server Startup Script Modifications

The Installer modifies the WebLogic Server startup script in order to add the newly installed libraries to the `CLASSPATH` as well as to add certain startup properties for the Java Virtual Machine.

Solaris and HP-UX CLASSPATH Modifications

The following new lines are added to the WebLogic Server startup script, above the line where CLASSPATH variable is defined:

```
AM_INSTALL_DIR=/opt/SUNWam/wlAgent

AM_SDK_DIR=$AM_INSTALL_DIR/amSDK
AM_SDK_LIB_DIR=$AM_SDK_DIR/lib
AM_SDK_LOCALE_DIR=$AM_SDK_DIR/locale
AM_SDK_JAR1=$AM_SDK_LIB_DIR/am_sdk.jar
AM_SDK_JAR2=$AM_SDK_LIB_DIR/am_services.jar
AM_SDK_JAR3=$AM_SDK_LIB_DIR/am_sso_provider.jar
AM_SDK_JARS=$AM_SDK_JAR1:$AM_SDK_JAR2:$AM_SDK_JAR3
AM_SDK_PATH1=$AM_SDK_DIR:$AM_SDK_LIB_DIR
AM_SDK_PATH2=$AM_SDK_LOCALE_DIR:$AM_SDK_JARS
AM_SDK_CLASSPATH=$AM_SDK_PATH1:$AM_SDK_PATH2

AM_AGT_DIR=$AM_INSTALL_DIR/amAgent
AM_AGT_CONFIG_DIR=$AM_AGT_DIR/config
AM_AGT_LOCALE_DIR=$AM_AGT_DIR/locale
AM_AGT_LIB_DIR=$AM_AGT_DIR/lib
AM_AGT_JAR1=$AM_AGT_LIB_DIR/amagent_core.jar
AM_AGT_JAR2=$AM_AGT_LIB_DIR/amagent_weblogic.jar
AM_AGT_JAR3=$AM_AGT_LIB_DIR/amagent_filter.jar
AM_AGT_JAR4=$AM_AGT_LIB_DIR/amagent_tools.jar
AM_AGT_JARS12=$AM_AGT_JAR1:$AM_AGT_JAR2
AM_AGT_JARS34=$AM_AGT_JAR3:$AM_AGT_JAR4
AM_AGT_JARS=$AM_AGT_JARS12:$AM_AGT_JARS34
AM_AGT_PATH1=$AM_AGT_DIR:$AM_AGT_CONFIG_DIR
AM_AGT_PATH2=$AM_AGT_LOCALE_DIR:$AM_AGT_LIB_DIR:$AM_AGT_JARS
AM_AGT_CLASSPATH=$AM_AGT_PATH1:$AM_AGT_PATH2

AM_CLASSPATH=$AM_SDK_CLASSPATH:$AM_AGT_CLASSPATH
```

Once these entries have been added, the AM_CLASSPATH is appended to the CLASSPATH variable.

```
CLASSPATH=$AM_CLASSPATH:$WL_HOME:$WL_HOME/lib/weblogic_sp.jar:$W
L_HOME/lib/weblogic.jar:$WL_HOME/samples/eval/cloudscape/lib/clo
udscape.jar:./config/examples/serverclasses
```

Windows CLASSPATH Modifications

The CLASSPATH is modified By adding the following lines to the WebLogic Server startup script immediately after the definition of the CLASSPATH variable.

```
set AM_INSTALL_DIR=C:\Sun\SUNWam\wlAgent

set AM_SDK_DIR=%AM_INSTALL_DIR%\amSDK
set AM_SDK_LIB_DIR=%AM_SDK_DIR%\lib
set AM_SDK_LOCALE_DIR=%AM_SDK_DIR%\locale
set AM_SDK_JAR1=%AM_SDK_LIB_DIR%\am_sdk.jar
set AM_SDK_JAR2=%AM_SDK_LIB_DIR%\am_services.jar
set AM_SDK_JAR3=%AM_SDK_LIB_DIR%\am_sso_provider.jar
set AM_SDK_JARS=%AM_SDK_JAR1%;%AM_SDK_JAR2%;%AM_SDK_JAR3%
set AM_SDK_PATH1=%AM_SDK_DIR%;%AM_SDK_LIB_DIR%
set AM_SDK_PATH2=%AM_SDK_LOCALE_DIR%;%AM_SDK_JARS%
set AM_SDK_CLASSPATH=%AM_SDK_PATH1%;%AM_SDK_PATH2%

set AM_AGT_DIR=%AM_INSTALL_DIR%\amAgent
set AM_AGT_CONFIG_DIR=%AM_AGT_DIR%\config
set AM_AGT_LOCALE_DIR=%AM_AGT_DIR%\locale
set AM_AGT_LIB_DIR=%AM_AGT_DIR%\lib
set AM_AGT_JAR1=%AM_AGT_LIB_DIR%\amagent_core.jar
set AM_AGT_JAR2=%AM_AGT_LIB_DIR%\amagent_weblogic.jar
set AM_AGT_JAR3=%AM_AGT_LIB_DIR%\amagent_filter.jar
set AM_AGT_JAR4=%AM_AGT_LIB_DIR%\amagent_tools.jar
set AM_AGT_JARS12=%AM_AGT_JAR1%;%AM_AGT_JAR2%
set AM_AGT_JARS34=%AM_AGT_JAR3%;%AM_AGT_JAR4%
set AM_AGT_JARS=%AM_AGT_JARS12%;%AM_AGT_JARS34%
set AM_AGT_PATH1=%AM_AGT_DIR%;%AM_AGT_CONFIG_DIR%
set
AM_AGT_PATH2=%AM_AGT_LOCALE_DIR%;%AM_AGT_LIB_DIR%;%AM_AGT_JARS%
set AM_AGT_CLASSPATH=%AM_AGT_PATH1%;%AM_AGT_PATH2%

set AM_CLASSPATH=%AM_SDK_CLASSPATH%;%AM_AGT_CLASSPATH%

set CLASSPATH=%AM_CLASSPATH%;%CLASSPATH%
```

The last line in the added text modifies the CLASSPATH to include the libraries provided by the Agent.

Adding Parameters to Java Virtual Machine

For the installation platforms Solaris 8, Windows 2000, and HP-UX 11 the following parameters are added to the Java Virtual Machine invocation command that loads the WebLogic Server:

```
-D"com.iplanet.coreservices.configpath=/opt/SUNWam/wlAgent/amSDK
/config/ums"
-D"max_conn_pool=10"
-D"min_conn_pool=1"
```

The resulting command on Solaris and HP-UX:

```
java $JAVA_OPTIONS -classpath $CLASSPATH
-Dweblogic.Domain=examples -Dweblogic.Name=examplesServer
-Dweblogic.management.password=$WLS_PW -Dbea.home=/bea
-Dcloudscape.system.home=./samples/eval/cloudscape/data
-Djava.security.policy==$WL_HOME/lib/weblogic.policy
-D"com.iplanet.coreservices.configpath=/opt/SUNWam/wlAgent/amSDK
/config/ums" -D"max_conn_pool=10" -D"min_conn_pool=1"
weblogic.Server
```

On Windows:

```
"%JAVA_HOME%\bin\java" -hotspot -ms64m -mx64m -classpath
"%CLASSPATH%" -Dweblogic.Domain=examples
-Dweblogic.Name=examplesServer
-Dweblogic.management.password=%WLS_PW% -Dbea.home="C:\bea"
-Dcloudscape.system.home=./samples/eval/cloudscape/data
-Djava.security.policy=="C:\bea\wlserver6.1\lib\weblogic.policy"
-D"com.iplanet.coreservices.configpath=C:/Sun/SUNWam/wlAgent/amS
DK/config/ums" -D"max_conn_pool=10" -D"min_conn_pool=1"
weblogic.Server
```

Installation of JCE 1.2.1 and JSSE 1.0.2 Extensions

The Installer also performs the installation of JCE 1.2.1 and JSSE 1.0.2 extensions which result in the modification of the file

`JAVA_HOME/jre/lib/security/java.security` and the copying of various jar files in the `JAVA_HOME/jre/lib/ext` directory. If the installation of any of these extensions fails, you can manually install them. For obtaining these extensions and documentation on how to install them, refer to the product website at:

<http://java.sun.com/products/jce> and

<http://java.sun.com/products/jsse>

WebSphere 4.0.4 AE

The Sun ONE Identity Server Policy Agent for WebSphere Server Installer performs certain configuration tasks. Depending on your system configuration and various other factors, these tasks may occasionally fail resulting in a unusable installation. Fortunately, such failures in most cases are recoverable by manually performing these tasks. The following sections describe how to configure Sun ONE Identity Server Policy Agent for WebSphere manually.

Modifications to Admin Server Configuration File

The following configuration changes are performed in the WebSphere Admin Server configuration file located at `WAS_root_dir/bin/admin.config`.

1. Update the property `com.ibm.ejs.sm.adminserver.classpath` and add all the SDK and AGENT related directories that contain property files.
2. Update the `-Dws.ext.dirs` value for `com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs`, to include SDK and AGENT related directories and jars.
3. Add the following arguments to `com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs` property:
 - o `Djava.protocol.handler.pkgs=com.ibm.net.ssl.internal.www.protocol`
 - o `Dcom.ipplanet.coreservices.configpath=<Agent_Install_Dir>/SUNWam/wasAgent/amSDK/config/ums`

Modifications to trustedserver.properties

4. Comment out all the lines that are not commented, and add the following lines to `WAS_root_dir/properties/trustedservers.properties` file:

```
com.ibm.websphere.security.trustassociation.enabled=true
com.ibm.websphere.security.trustassociation.types=amagent
com.ibm.websphere.security.trustassociation.amagent.interceptor=com
.sun.amagent.websphere.interceptor.AgentInterceptor
com.ibm.websphere.security.trustassociation.amagent.config=AMAgent
```

Modifications to sas.client.props

5. Update the following properties in the file:
`WAS_root_dir/properties/sas.client.props`

```
com.ibm.CORBA.securityEnabled = true
com.ibm.CORBA.loginSource = properties
com.ibm.CORBA.loginUserId = <Realm Administrator>
com.ibm.CORBA.loginPassword = <Realm Administrator Password>
com.ibm.CORBA.principalName = <Realm Name>/<Realm Administrator>
```

6. Restart the Administration Server and the Application Server instance.

NOTE Before performing Agent Realm configuration, the Administration Server and Application server instance must be restarted, so that the above modifications take effect. During the Agent Realm installation, the `authenticate()` method of Realm is invoked with the given Realm Administrator credentials. The SDK and the Agent related classes must be in the classpath. The realm is installed only when the authentication is successful.

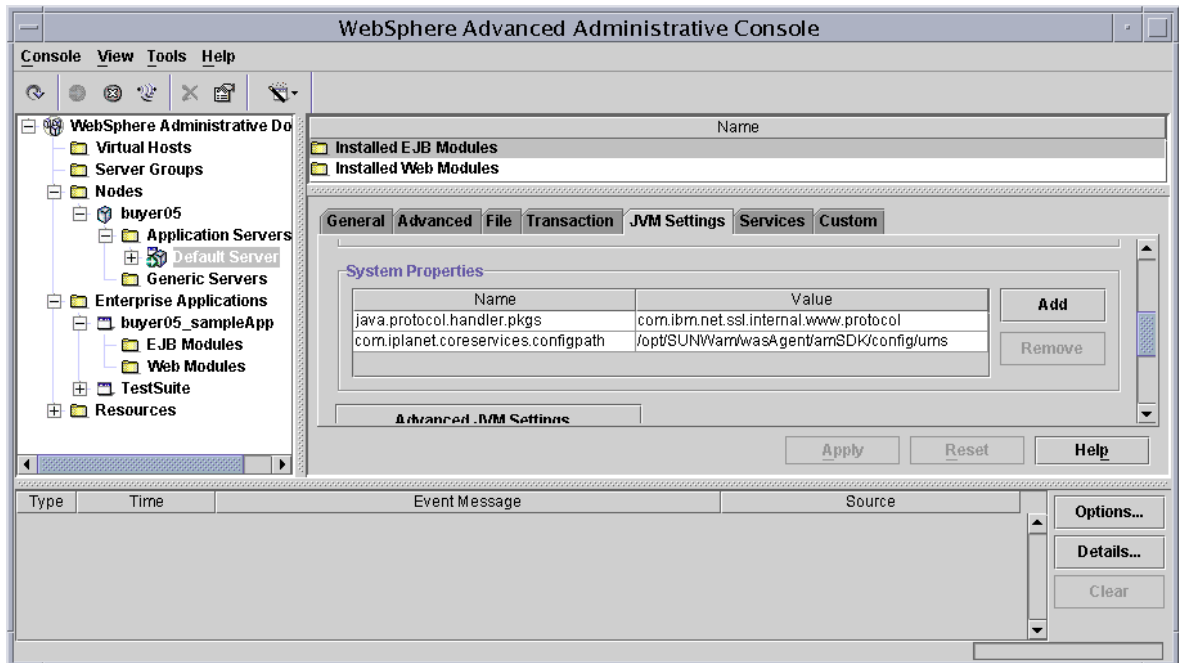
Configurations Through Administrative Console

Setting System Properties for the Application Server Instance

The following properties must be added to each Application server instance, through the Administrative console as shown in Figure A-1.

- `Djava.protocol.handler.pkgs=com.ibm.net.ssl.internal.www.protocol`
- `Dcom.iplanet.coreservices.configpath=`
`Agent_Install_Dir/SUNWam/wasAgent/amSDK/config/ums`

Figure A-1 Setting System Properties



Agent Realm Configuration

The Agent Realm can be configured from the Security Center of Administrative Console. Follow the steps given below:

1. Start the WebSphere Administrative console using the following command:
`# WAS_roor_dir/bin/adminclient.sh`
2. In the Administrative Console window, choose Console > Security Center.
3. In the Security Center window, click on Authentication tab. See Figure A-2.

4. Choose Lightweight Third Party Authentication (LTPA) for authentication mechanism.
5. Choose Custom Registry Option.
6. Enter the Administrator's user ID for Security Server Id. For example, amAdmin for o=siroe.com.
7. Enter the password for the Administrative User for "Security Server Password."
8. Enter the value `com.sun.amagent.websphere.realm.AgentRealm` for Custom Registry Class.
9. Click on check box Enable Web trust association.
10. Click Apply. A message "Changes will take effect only after Administration Server is restarted" will be displayed. Click OK
11. Click on General tab and enable the check box Enable Security.
12. Stop all Application Server instances.
13. Stop the Administration Server.
14. Restart Administration Server and Application Server instances for changes to take effect.

Figure A-2 Agent Realm Configuration

Security Center

General Authentication Role Mapping Run As Role Mapping Administrative Role

Authentication Mechanism: ☐ Local Operating System ☒ Lightweight Third Party Authentication (LTPA)

LTPA Settings

* **Token Expiration:** 120 minutes

☒ **Enable Single Sign On (SSO)**

* **Domain:** eng.siroe.com

☐ **Limit to SSL connections only**

☒ **Enable Web trust association**

Generate Keys... Import Key... Export Key...

☐ **LDAP** ☒ **Custom User Registry**

Custom User Registry Settings

A Custom User Registry is a user defined registry. Custom user registries are defined by implementing the com.ibm.websphere.security.CustomRegistry interface. The CustomRegistry interface and a sample implementation can be found in the security section of the WebSphere documentation.

* **Security Server ID:** arnldapuser

* **Security Server Password:** *****

* **Custom User Registry Classname:** agent.websphere.realm.AgentRealm

Special Custom Settings

OK Cancel Apply Help

Sun ONE Application Server 7.0

The following sections describe the configuration tasks performed by Sun ONE Identity Server Policy Agent Installer.

Application Server Config Files

The following configuration files are modified by the Installer:

SIAS_Install_Dir/SUNWappserver7/domains/domain1/*server-instance*/config/server.xml

SIAS_Install_Dir/SUNWappserver7/domains/domain1/server-instance/config/login.conf

SIAS_Install_Dir/SUNWappserver7/domains/domain1/server-instance/config/server.policy

Modifications in server.xml

Installer modifies `server.xml` in order to add all the newly installed libraries to the `classpath` as well as to add certain properties for the Java Virtual Machine. Also, it adds Agent Realm and makes it default to be used.

classpath Modifications

The `classpath` is modified by adding the following lines to the Application Server startup script immediately after the definition of the `classpath` variable.

```
Agent_Install_Dir/SUNWam/asAgent/amSDK;  
Agent_Install_Dir/SUNWam/asAgent/amSDK/lib;  
Agent_Install_Dir/SUNWam/asAgent/amSDK/locale;  
Agent_Install_Dir/SUNWam/asAgent/amSDK/lib/am_sdk.jar;  
Agent_Install_Dir/SUNWam/asAgent/amSDK/lib/am_services.jar;  
Agent_Install_Dir/SUNWam/asAgent/amSDK/lib/am_sso_provider.jar;  
Agent_Install_Dir/SUNWam/asAgent/amAgent;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/config;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/locale;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/lib;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/lib/amagent_core.jar;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/lib/amagent_as.jar;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/lib/amagent_filter.jar;  
Agent_Install_Dir/SUNWam/asAgent/amAgent/lib/amagent_tools.jar
```

You can verify the modifications by invoking the Admin Console:

1. Click the *server-instance* under Application Server Instances.
2. On the right pane, click JVM Settings > Path Settings.
3. Check the values of Classpath Suffix, which includes the above values.

Adding Parameters to Java Virtual Machine

The following parameters are added to the Java Virtual Machine invocation command that loads the Application Server:

```
-Dcom.iplanet.coreservices.configpath=Agent_Install_Dir/SUNWam/asAgent/amSDK/config/ums
-Dmax_conn_pool=10
-Dmin_conn_pool=1
```

Adding Agent Realm

The following lines are added under `<security-service>` element:

```
<auth-realm name="agentRealm"
classname="com.sun.amagent.as.realm.AgentRealm">
<property name="jaas-context" value="agentRealm"/>
</auth-realm>
```

Here `name=agentRealm` is value provided at installation time.

You can verify this using the Admin Console.

- Choose Security > Realms and verify the Agent Realm.

Making Agent Realm as Default

`default-realm` attribute of `<security-service>` element is modified as follows:

```
<security-service default-realm="agentRealm" anonymous-role="ANYONE"
audit-enabled="false">
```

You can verify this using the Admin Console.

- Click Security and verify the value of Default Realm.

Modifications in login.conf

Installer modifies `login.conf` to define `LoginModule` for the `jaas-context` as follows:

```
agentRealm {  
    com.sun.amagent.as.realm.AgentLoginModule required;  
};
```

Modifications in server.policy

Installer modifies `server.policy` to give Programmatic Login permission to `agent-filter.jar` as follows:

```
// ProgrammaticLoginPermission to Sun(TM) ONE Identity Server  
Policy Agent Filter  
grant codeBase  
"file:Agent_Install_Dir/SUNWam/asAgent/amAgent/lib/amagent_filter.jar  
{  
    permission  
com.sun.appserv.security.ProgrammaticLoginPermission "login";  
};
```

Sample Scenarios for Role-to-Principal Mapping

This appendix provides sample scenarios for creating role-to-principal mappings in the following application servers:

- WebLogic 6.1 SP2
- WebSphere 4.0.4 AE
- Sun ONE Application Server 7.0

WebLogic 6.1 SP2

Declarative Security

Consider an Enterprise JavaBeans component that must have protected access for one of its methods. This can be enforced by adding a security-role and method-permission element in the assembly-descriptor element of the `ejb-jar.xml` deployment descriptor.

```
<?xml version="1.0"?>

<!DOCTYPE ejb-jar PUBLIC
'-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 2.0//EN'
'http://java.sun.com/dtd/ejb-jar_2_0.dtd'>
<ejb-jar>
  ...
  <assembly-descriptor>
    <security-role>
      <role-name>FOO</role-name>
    </security-role>
```

```

    <method-permission>
      <role-name>FOO</role-name>
    <method>
      <ejb-name>WebProxy</ejb-name>
      <method-intf>Remote</method-intf>
      <method-name>getWebPage</method-name>
      <method-params>
        <method-param>java.lang.String</method-param>
        <method-param>java.lang.String</method-param>
      </method-params>
    </method>
  </method-permission>
</assembly-descriptor>
</ejb-jar>

```

This security role FOO can be mapped to a real principal using the weblogic-ejb-jar.xml deployment descriptor.

```

<?xml version="1.0"?>
<!DOCTYPE weblogic-ejb-jar PUBLIC "-//BEA Systems, Inc.//DTD WebLogic 6.0.0 EJB//EN"
'http://www.bea.com/servers/wls6000/dtd/weblogic-ejb-jar.dtd'>
<weblogic-ejb-jar>
  <weblogic-enterprise-bean>
    <ejb-name>WebProxy</ejb-name>
    <jndi-name>ejb.WebProxy</jndi-name>
  </weblogic-enterprise-bean>
  <security-role-assignment>
    <role-name>FOO</role-name>
    <principal-name>amAdmin</principal-name>
  </security-role-assignment>
</weblogic-ejb-jar>

```

Programmatic Security

Consider a sample application in which one of the Servlets uses the programmatic security API such as `HttpServletRequest.isUserInRole(String)`. Assuming that the role name used within the Servlet code is `SAMPLE-ROLE`, its deployment descriptor.

```

<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
  <display-name>Sample Security Aware application</display-name>

```



```

    <filter>
      <filter-name>Agent</filter-name>
      <display-name>Agent</display-name>
      <description>Identity Server Policy Agent for WebLogic
Server 6.1</description>
      <filter-class>
        com.ipplanet.amagent.weblogic.filter.AgentFilter
      </filter-class>
    </filter>
    <filter-mapping>
      <filter-name>Agent</filter-name>
      <url-pattern>/*</url-pattern>
    </filter-mapping>
    <servlet>
      <servlet-name>SampleServlet</servlet-name>
      <servlet-class>
        com.ipplanet.sample.SampleServlet
      </servlet-class>
      <security-role-ref>
        <description>
          This role is a sample test role for the
          security aware servlet
        </description>
        <role-name>SAMPLE-ROLE</role-name>
        <role-link>SAMPLE-ROLE-LINK</role-link>
      </security-role-ref>
    </servlet>
    <servlet-mapping>
      <servlet-name>SampleServlet</servlet-name>
      <url-pattern>/Test</url-pattern>
    </servlet-mapping>
    <security-role>
      <description>Some description</description>
      <role-name>SAMPLE-ROLE-LINK</role-name>
    </security-role>
  </web-app>

```

The SAMPLE-ROLE-LINK is mapped to an actual principal using the weblogic.xml deployment descriptor.

```

<!DOCTYPE weblogic-web-app PUBLIC "-//BEA Systems, Inc.//DTD Web
Application 6.1//EN" "http://www.be
a.com/servers/wls610/dtd/weblogic-web-jar.dtd">
<weblogic-web-app>
  <security-role-assignment>
    <role-name>SAMPLE-ROLE-LINK</role-name>
    <principal-name>Employee</principal-name>
  </security-role-assignment>
</weblogic-web-app>

```

WebSphere 4.0.4 AE

Web Authorization

1. In `web.xml`, there is an element `security-constraint` which is used to define constraints placed on various parts of this web application.

```

?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.2//EN"
"http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
<web-app id="WebApp_1">
  <display-name>Test Application</display-name>
  <description>A Web Application for Testing Various Test Cases for
WebSphere Agent</description>
  .....
  .....
  <servlet id="Servlet_10">
    <servlet-name>ProtectedServlet</servlet-name>
    <display-name>ProtectedServlet</display-name>
    <description>A Protected servlet that accesses an unprotected
EJB</description>
    <servlet-class>com.agent.servlet.ProtectedServlet</ser
vlet-class>
  </servlet>

  <servlet-mapping id="ServletMapping_10">
    <servlet-name>ProtectedServlet</servlet-name>
    <url-pattern>/ProtectedServlet</url-pattern>
  </servlet-mapping>

  <security-constraint id="SecurityConstraint_2">
    <web-resource-collection id="WebResourceCollection_2">
      <web-resource-name>Protected</web-resource-name>
      <url-pattern>/ProtectedServlet</url-pattern>
      <http-method>GET</http-method>
      <http-method>POST</http-method>
    </web-resource-collection>
    <auth-constraint id="AuthConstraint_2">
      <description>Protected-tc4:+:</description>
      <role-name>Protected-tc3</role-name>
    </auth-constraint>
    <user-data-constraint id="UserDataConstraint_2">
      <transport-guarantee>NONE</transport-guarantee>
    </user-data-constraint>
  </security-constraint>

  <security-role id="SecurityRole_5">
    <role-name>Protected-tc3</role-name>
  </security-role>

```

```

.....
.....
</web-app>

```

2. In the `ibm-application-bnd.xml` file, define role mappings for the role `Protected-tc3` as:

```

<applicationbnd:ApplicationBinding xmi:version="2.0" xmlns:xmi=
"http://www.omg.org/XMI"
xmlns:applicationbnd="applicationbnd.xmi"
  xmlns:application="application.xmi" xmlns:common="common.xmi"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmi:id=
"Application_ID_Bnd">

  <appName xsi:nil="true"/>
  <application href="META-INF/application.xml#Application_ID"/>
  <authorizationTable xmi:id="AuthorizationTable_1">
    .....
    .....
    <authorizations xmi:id="RoleAssignment_4">
      <role href="META-INF/application.xml#SecurityRole_5"/>
      <users xmi:id="User_1" name="amAdmin"/>
      <groups xmi:id="Group_4" name="manager"/>
    </authorizations>
    .....
    .....
  </authorizationTable>
  <runAsMap xmi:id="RunAsMap_1"/>
</applicationbnd:ApplicationBinding>

```

EJB Authorization

Consider an Enterprise JavaBeans component that must have protected access for one of its methods. This can be enforced by adding a security-role and method-permission element in the assembly-descriptor element of the `ejb-jar.xml` deployment descriptor.

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD
Enterprise JavaBeans 1.1//EN"
"http://java.sun.com/j2ee/dtds/ejb-jar_1_1.dtd">
<ejb-jar id="ejb-jar_ID">
  <enterprise-beans>

```

```

.....
.....
<session id="Session_3">
  <ejb-name>ProtectedEJB</ejb-name>
  <home>com.agent.ejb.ProtectedEJBHome</home>
  <remote>com.agent.ejb.ProtectedEJB</remote>
  <ejb-class>com.agent.ejb.ProtectedEJBImpl</ejb-class>
  <session-type>Stateless</session-type>
  <transaction-type>Container</transaction-type>
</session>
.....
.....
</enterprise-beans>

<assembly-descriptor id="AssemblyDescriptor_ID">
  .....
  .....
  <security-role id="SecurityRole_4">
    <role-name>Protected-tc2</role-name>
  </security-role>
  .....
  .....
  <method-permission id="MethodPermission_2">
    <description>tc2:::</description>
    <role-name>Protected-tc2</role-name>
    <method id="MethodElement_2">
      <ejb-name>ProtectedEJB</ejb-name>
      <method-intf>Remote</method-intf>
      <method-name>protectedMethod</method-name>
      <method-params></method-params>
    </method>
  </method-permission>

  .....
  ....
</assembly-descriptor>
</ejb-jar>

```

3. In the `ibm-application-bnd.xml` file, define role mappings for the role `Protected-tc2`:

```

<applicationbnd:ApplicationBinding xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns:applicationbnd="applicationbnd.xml"
  xmlns:application="application.xml" xmlns:common="common.xml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmi:id="Application_ID_Bnd">

  <appName xsi:nil="true"/>
  <application href="META-INF/application.xml#Application_ID"/>
  <authorizationTable xmi:id="AuthorizationTable_1">

```

```

.....
.....
<authorizations xmi:id="RoleAssignment_3">
  <role href="META-INF/application.xml#SecurityRole_4"/>
  <groups xmi:id="Group_2" name="manager"/>
  <groups xmi:id="Group_3" name="employee"/>
</authorizations>
.....
.....
</authorizationTable>
<runAsMap xmi:id="RunAsMap_1"/>
</applicationbnd:ApplicationBinding>

```

Sun ONE Application Server 7.0

Declarative Security

Consider an Enterprise JavaBeans component that must have protected access for one of its methods. This can be enforced by adding a security-role and method-permission element in the assembly-descriptor element of the `ejb-jar.xml` deployment descriptor.

```

<?xml version="1.0"?>

<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise
JavaBeans 2.0//EN"
'http://java.sun.com/dtd/ejb-jar_2_0.dtd'>
<ejb-jar>
  ...
  <assembly-descriptor>
    <security-role>
      <role-name>staffRole</role-name>
    </security-role>
    <method-permission>
      <role-name>staffRole</role-name>
      <method>
        <ejb-name>ConverterEJB</ejb-name>
        <method-name>dollarToYen</method-name>
      </method>
    </method-permission>
  </assembly-descriptor>
  ...
</ejb-jar>

```

This security role `staffRole` can then be mapped to a real principal in `sun-application.xml`.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE sun-application PUBLIC "-//Sun Microsystems, Inc.//DTD
Sun ONE Application Server 7.0 J2EE Application 1.3//EN"
'http://www.sun.com/software/sunone/appserver/dtds/sun-applicati
on_1_3-0.dtd'>

<sun-application>

  <security-role-mapping>
    <role-name>staffRole</role-name>
    <group-name>staff</group-name>
    <principal-name>amAdmin</principal-name>
  </security-role-mapping>

</sun-application>
```

NOTE For stand-alone ejb modules there is no `sun-application.xml`, in that case this data goes into `sun-ejb-jar.xml`.

Programmatic Security

Consider a sample application in which one of the Servlets uses the programmatic security API such as `HttpServletRequest.isUserInRole(String)`. Assuming that the role name used within the Servlet code is `SAMPLE-ROLE`, its deployment descriptor.

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.3//EN" 'http://java.sun.com/dtd/web-app_2_3.dtd'>
<web-app>
  ...
  <servlet>
    <servlet-name>ProtectedServlet</servlet-name>
    <servlet-class>
      com.sun.sample.ProtectedServlet
    </servlet-class>
    <security-role-ref>
      <description>
        This role is a sample test role for the
        security aware servlet
      </description>
    </security-role-ref>
  </servlet>
</web-app>
```

```

        </description>
        <role-name>SAMPLE-ROLE</role-name>
        <role-link>SAMPLE-ROLE-LINK</role-link>
    </security-role-ref>
</servlet>
<servlet-mapping>
    <servlet-name>ProtectedServlet</servlet-name>
    <url-pattern>/ProtectedServlet</url-pattern>
</servlet-mapping>
...
<security-constraint>
    <web-resource-collection>
        <web-resource-name>basic security test</web-resource-name>
        <url-pattern>/ProtectedServlet</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <role-name>SAMPLE-ROLE-LINK</role-name>
    </auth-constraint>
</security-constraint>
...
<security-role>
    <description>Some description</description>
    <role-name>SAMPLE-ROLE-LINK</role-name>
</security-role>
...
</web-app>

```

In the `sun-application.xml` file, define role mappings as:

```

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE sun-application PUBLIC "-//Sun Microsystems, Inc.//DTD
Sun ONE Application Server 7.0 J2EE Application 1.3//EN"
'http://www.sun.com/software/sunone/appserver/dtds/sun-applicati
on_1_3-0.dtd'>

<sun-application>

    <security-role-mapping>
        <role-name>SAMPLE-ROLE-LINK</role-name>
        <group-name>staff</group-name>
        <principal-name>amAdmin</principal-name>
    </security-role-mapping>

</sun-application>

```


Using the Policy Agent Debug Engine

Sun ONE Identity Server Policy Agent for Application Servers is equipped with a sophisticated Debug Engine that can be used to gather statistics about your deployment, oversee the execution of the Agent as it protects various applications and troubleshoot hard to locate problems with your installation.

To use this engine, set the properties correctly for the Debug Engine configuration.

A few things worth noting are:

- When the Debug Engine is operational, that is the debug level is set to a value other than 0, the Agent reports varying set of information to the Debug Engine. The Debug Engine has to process this information in order to store it or display it appropriately. This process results in a overhead that should be avoided if the performance of the system is of utmost significance. Therefore, the Debug Engine should not be used when the system is in production. It should be used during testing of the deployed application under controlled environment.
- If the Debug Engine is turned on, that is the debug level is set to a value other than 0, it is possible that Print STDOUT flag is disabled as well as the debug file is not specified correctly. In such a situation, all the debug messages are lost, yet the performance of the system suffers. Therefore, to ensure that the debug engine is turned off, set the debug level to 0.
- The Debug Engine's date and time format is configurable and can be used to generate specific strings which may help distinguish between Agent Messages from other messages being generated on the Application Server console.

Index

A

- Agent Filter 138
- Agent Realm 138
- Agents work
 - Web and Web Proxy Server 19
 - WebLogic 138
- AMAgent.properties 29

C

- Certificate Server
 - Documentation 14
- Configuring
 - Domino DSAPI Filter 52
 - Domino DSAPI Filter for Multiple Server Partitions 77
 - IBM HTTP Server 52
- configuring
 - Apache Web Server 114
 - WebLogic Agent 166
 - WebLogic Server 155
- configuring SSL 53
 - web servers
 - Linux 7.2 125
 - Solaris 8 53

D

- Developer Information 15
- Documentation
 - Certificate Server 14
 - Proxy Server 15
 - Web Server 14
- Domino
 - Configuring DSAPI Filter for Multiple Server Partitions 77
- Domino DSAPI Filter
 - Configuring 52
- Downloads
 - Sun ONE Software 15

F

- failover protection 24
- FQDN 31

G

- global not-enforced IP Address list 26
- global not-enforced URL list 25

I

- IBM HTTP Server 37
 - Configuring 52
- Identity Server
 - Related Product Information 14
- installation script 37
- Installing
 - Sun ONE Application Server 7.0 Policy Agent 252
- installing
 - Apache 1.3.26 Agent 115
 - Microsoft IIS 4.0 Agent 94, 98
 - Microsoft IIS 5.0 Agent 70, 74
 - Proxy Server Agent 41
 - Web Server Agent 37, 43
 - WebLogic Agent 145

J

- JRE requirement 23

L

- Lotus Domino 5.0.11
 - installation script 37

O

- overview 19

P

- pre-installation tasks 139
- Professional Services 15
- Proxy Server
 - Documentation 15

- Proxy Server Agent 41

R

- remote web server 23

S

- Silent Installation
 - Sun ONE Application Server 7.0 Policy Agent 270
- silent installation 49
- Solaris
 - Patches 15
 - Support 15
- Sun ONE
 - Support 15
- Sun ONE Application Server 7.0 Policy Agent
 - Installing 252
 - Silent Installation 270
 - Supported Platforms 251
- Support
 - Professional Services 15
 - Solaris 15
 - Sun ONE 15
- Supported Platforms
 - Sun ONE Application Server 7.0 Policy Agent 251
- Supported Servers 21
 - Red Hat Linux 7.2 22
 - Solaris 8 21
 - Solaris 9 21
 - Windows 2000 22
 - Windows NT 4.0 22

U

- uninstalling
 - Apache 1.3.26 Agent 118

- Microsoft IIS 4.0 97
- Microsoft IIS 5.0 83
- Web Server Agent 62, 85
- WebLogic Agent 187
- updating Agent Cache 24
 - cache 24
 - hybrid cache 25

V

- Verifying installation 33

W

- Web Authorization
 - Websphere 322
- Web Server
 - Documentation 14
- WebLogic Agent 138, 252
- WebLogic Server
 - launching
 - HP-UX 11 143
 - Solaris 8 140
 - Windows 2000 Server 142
- Websphere
 - Web Authorization 322
- WebSphere 4.0.4 AE
 - Configuration 311

