IBM WebSphere Commerce

Security Guide

Version 5.5

IBM WebSphere Commerce

Security Guide

Version 5.5

Note:
Before using this information and the product it supports, be sure to read the information in "Notices" on page 215.

First Edition, Second Revision (January 2004)

This edition applies to IBM WebSphere Commerce Version 5.5 (product number 5724-A18) and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality.

IBM welcomes your comments. You can send your comments by using the online IBM WebSphere Commerce documentation feedback form, available at the following URL:

http://www.ibm.com/software/commerce/rcf.html

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

About this book

This document describes the security features of WebSphere Commerce and how to configure these features.

It details WebSphere Commerce security issues and features such as authentication, authorization, and access control policies. The objective of this document is to provide the persons responsible for security at your site (which likely includes a system administrator or WebSphere Commerce site administrator) with a comprehensive document to enable them to reliably secure a WebSphere Commerce production site.

The intended audience for this document is the chief security officer or the security administrator for a WebSphere Commerce site.

Important

This document covers only WebSphere Commerce security issues related to deploying an e-commerce site. Issues relating to vulnerabilities of your operating system are not covered. You should consult with your operating system vendor to determine the appropriate measures that you should take to secure your operating system.

Summary of changes

This Security Guide, and any updated versions of this Security Guide, will be available in the WebSphere® Commerce Technical Library Web page (http://www.ibm.com/software/commerce/library/). For additional information for your WebSphere Commerce edition, see the overview pages:

- Business Edition (`http://www.ibm.com/software/webservers/commerce/wc_be/`)
- Professional Edition (`http://www.ibm.com/software/commerce/wscom/support/wc_pe/`)

For additional support information, see the WebSphere Commerce Support page (`http://www.ibm.com/software/commerce/support/`).

To learn about last-minute changes to the product, see the updated product README file, also available from the above Web site.

Updates from the last version of this document are identified by revision characters contained in the margin. This book uses the following conventions for revision characters:

- The "+" character identifies updates that have been made in the current version of this document.
- The "|" character identifies any updates that have been made in the previous versions of this document.

| The following table shows the main changes that have been made to this book.

Change	Chapter or page affected
Change in behavior made to access logging.	"Enabling access logging" on page 58.

Change	Chapter or page affected
Consideration added for how passwords are stored by LDAP servers.	"LDAP server password storage consideration" on page 76.
Correction to the section on Enabling WebSphere Application Server Security.	Chapter 16, "Enabling WebSphere Application Server security," on page 173, in particular "Enabling security with an LDAP user registry" on page 174.
Added important note on WebSphere Application Server default test certifcates.	Chapter 16, "Enabling WebSphere Application Server security," on page 173.
+ + Added information on configuring SSL Accelerator.	"Enabling the SSL Accelerator option" on page 192.

Navigating through this book

This document is divided into the following parts:

- Part 1, "WebSphere Commerce security concepts," on page 1 discusses the WebSphere Commerce security model and provides a conceptual overview of WebSphere Commerce security. This part will be of interest to anyone that wants a general overview of WebSphere Commerce security or to plan for security at a WebSphere Commerce site.
- Part 2, "Administering security authentication," on page 47 discusses WebSphere Commerce administration tasks pertaining to site security. This part will be of interest to anyone performing administration tasks pertaining to site security.
- Part 3, "Administering security authorization," on page 83 discusses WebSphere Commerce authorization tasks pertaining to access control. This part will be of interest to anyone performing system authorization tasks pertaining to access control on WebSphere Commerce.
- Part 4, "Payments security," on page 165 discusses WebSphere Commerce administration tasks pertaining to WebSphere Commerce Payments security. This part will be of interest to anyone administering WebSphere Commerce Payments.
- Part 5, "Miscellaneous security topics," on page 171 discusses miscellaneous WebSphere Commerce system administration tasks such as enhancing WebSphere Application Server security. This part will be of interest to system administrators responsible for security.

Conventions used in this book

This book uses the following highlighting conventions:

Boldface type	Indicates commands or graphical user interface (GUI) controls such as names of fields, icons, or menu choices.
`Monospace type`	Indicates examples of text you enter exactly as shown, file names, and directory paths and names.
Italic type	Used to emphasize words. Italics also indicate names for which you must substitute the appropriate values for your system.
host_name	The fully qualified host name of your WebSphere Commerce server (for example, `server.mydomain.ibm.com` is fully qualified).
instance_name	The name of the WebSphere Commerce instance with which you are working.

Windows *drive*	The letter representing the drive on which you installed the product or component being discussed (for example, `C:`).

This icon marks a Tip - additional information that can help you complete a task.

Important
These sections highlight especially important information.

Attention
These sections highlight information intended to protect your data.

Business indicates information specific to WebSphere Commerce Business Edition.

Professional indicates information specific to WebSphere Commerce Professional Edition.

AIX indicates information specific to WebSphere Commerce for AIX®.

400 indicates information specific to WebSphere Commerce for the IBM® @server iSeries™ 400® (formerly called AS/400®)

Linux indicates information specific to WebSphere Commerce for Linux.

Solaris indicates information specific to WebSphere Commerce for Solaris Operating Environment software.

Windows indicates information specific to WebSphere Commerce for Windows® 2000.

Path variables

This Guide uses the following variables to represent directory paths:

DB2_installdir
This variable represents the actual installation directory for DB2 Universal Database on your machine. The following are the default installation directories for DB2 Universal Database on various operating systems:

AIX	`/usr/lpp/db2_08_01`
400	Not applicable (installed as part of operating system)
Linux	`/opt/IBM/db2/V8.1`
Solaris	`/opt/IBM/db2/V8.1`
Windows	`C:\Program Files\WebSphere\sqllib`

HTTPServer_installdir

This variable represents the actual installation directory for IBM HTTP Server on your machine. The following are the default installation directories IBM HTTP Server on various operating systems:

AIX	`/usr/IBMHttpServer`
400	Not applicable (installed as part of operating system)
Linux	`/opt/IBMHttpServer`
Solaris	`/opt/IBMHttpServer`
Windows	`C:\Program Files\WebSphere\IBMHTTPServer`

Oracle_installdir

This variable represents the actual installation directory for Oracle on your machine. The following are the default installation directories Oracle on various operating systems:

AIX	`/oracle/u01/app/oracle/product/9.2.0`
400	Not applicable for OS/400®.
Linux	Not applicable for Linux
Solaris	`/opt/oracle/u01/app/oracle/product/9.2.0`
Windows	`C:\oracle\ora91`

WAS_installdir

This variable represents the actual installation directory for WebSphere Application Server on your machine. The following are the default installation directories for WebSphere Application Server on various operating systems:

AIX	`/usr/WebSphere/AppServer`
400	`/QIBM/ProdData/WebAS5/Base`
Linux	`/opt/WebSphere/AppServer`
Solaris	`/opt/WebSphere/AppServer`
Windows	`C:\Program Files\WebSphere\AppServer`

WAS_userdir

400 This variable represents the directory for all the data that is used by WebSphere Application Server which can be modified or needs to be configured by the user, on an iSeries machine. The default for this directory is:

400	`/QIBM/UserData/WebAS5/Base/`*`WAS_instance_name`*

WC_installdir

This variable represents the actual installation directory for WebSphere Commerce on your machine. The following are the default installation directories for WebSphere Commerce on various operating systems:

AIX	`/usr/WebSphere/CommerceServer55`
400	`/QIBM/ProdData/CommerceServer55`
Linux	`/opt/WebSphere/CommerceServer55`
Solaris	`/opt/WebSphere/CommerceServer55`
Windows	`C:\Program Files\WebSphere\CommerceServer55`

WC_userdir

400 This variable represents the directory for all the data that is used by WebSphere Commerce which can be modified or needs to be configured by the user on an iSeries system. The default for this directory is:

400	`/QIBM/UserData/CommerceServer55`

Part 1. WebSphere Commerce security concepts

This part provides a conceptual overview of WebSphere Commerce security.

Chapter 1. Introduction to the WebSphere Commerce security model

This chapter describes the WebSphere Commerce security model as well as various WebSphere Commerce security concepts.

Overview

The information in this document describes the notions of authentication, authorization, policies, and confidentiality:

What is authentication?

Authentication is the process of verifying that users or applications are who they claim to be. In a WebSphere Commerce system, authentication is required for all users and applications accessing the system, with the exception of guest users. The user authentication process is always performed under SSL. This ensures that a third party using network-sniffing programs cannot *snoop* on the network when a user submits a password. Passwords are never decrypted during the authentication process, as is the common security practice. All user passwords are one-way hashed and encrypted using a 128–bit key, known as the *merchant key*. The merchant key is specified during installation and configuration of the WebSphere Commerce system.

The WebSphere Commerce system has its own passwords for administration purposes. These passwords should periodically be changed as part of a WebSphere Commerce site-wide security policy. For details how to change the WebSphere Commerce system passwords, see Chapter 6, “Setting and changing passwords,” on page 71.

What is authorization?

Authorization is the process of determining whether a user can perform a specific operation on a resource. Authorization is determined from the access control policies governing WebSphere Commerce resources. In a WebSphere Commerce system, access control is needed in two areas:

- To protect the WebSphere Commerce Enterprise JavaBeans™ (EJB beans) from unauthorized access. This process is discussed in Chapter 16, “Enabling WebSphere Application Server security,” on page 173.
- To ensure that only authorized parties can execute different groups of WebSphere Commerce commands. This process is discussed in the section on "Access Control" in the *WebSphere Commerce Programming Guide and Tutorials* document.

What are access control policies?

Assuming that you have finished defining the organizations and users that will participate in your e-commerce site, you can now manage their activities through a set of policies, a process referred to as *access control*.

An access control policy is a rule that describes which group of users is authorized to perform particular activities on your site. These activities can range from registration, to managing auctions, to updating the product catalog, and granting approvals on orders, as well as any of the hundreds of other activities that are required to operate and maintain an e-commerce site.

The policies are what grants users access to your site. Unless they are authorized to perform their responsibilities through one or more access control policies, users have no access to any of your site’s functions.

The authorization model for WebSphere Commerce is based upon the enforcement of access control policies. Access control policies are enforced by the access control Policy Manager. In general, when a user attempts to access a protectable resource, the access control policy manager first determines what access control policies are applicable for that user and then, based upon the applicable access control policies, it determines if the user is allowed to perform the requested operation on the given resource.

What is an audit trail?

In computing, an *audit trail* is used to refer to electronic or paper logs that are used to track computer activity. For example, an employee might have access to a portion of a corporate network such as account receivable, but may not be authorized to access other portions of the system, such as payroll. If that employee attempts to access an unauthorized section by typing in passwords, this improper activity is recorded in the audit trail.

In e-commerce systems, audit trails are used to record customer activity. An audit trail records a customer’s initial contact with the system as well as subsequent actions such as payment and delivery of the product or service. Companies can use the audit trail to respond to any inquiries or complaints. It can also use the audit trail to reconcile accounts, to provide analysis and historical information for future planning and budgeting, and to provide a record of sales in case of a tax audit.

Audit trails can also be used to investigate computer crimes over cyberspace and the internet. To expose an individual conducting malicious attacks on a system, investigators can follow the audit trail left by the perpetrator. Sometimes the perpetrators of cyber crimes unknowingly leave behind audit trails in activity logs with their internet service providers or perhaps through chat room logs.

What is confidentiality?

Confidentiality is the process of protecting sensitive information from being deciphered by unintended recipients. In the WebSphere Commerce system, confidentiality is required when sensitive information flows from the user’s browser to the WebSphere Commerce server, and back from the WebSphere Commerce server to the user’s browser. As discussed in Chapter 17, “Enabling SSL for production with IBM HTTP Server,” on page 185, using Secure Sockets Layer (SSL), provides confidentiality for this scenario.

Confidentiality is also a strong requirement in the area of session management. Because the Hypertext Transfer Protocol (HTTP) protocol is state less, a *cookie* is commonly used to continuously identify the user to the WebSphere Commerce server. If this cookie is stolen, then the user account can be compromised. This is commonly known as *session hijacking*. WebSphere Commerce prevents session hijacking by using unique features of the cookie specifications as discussed in Chapter 5, “Session management,” on page 65.

General security considerations

Ongoing security assessment

The WebSphere Commerce product lines normally undergo security analysis from an independent group of IBM Security experts. These experts perform security analysis from the point of view of a user with only access to WebSphere Commerce through a browser to the more privileged users that have an account on the same system that WebSphere Commerce server is running. The feedback from the security experts' analysis is used to continually improve the security of WebSphere Commerce.

Security improvements in WebSphere Commerce 5.5

WebSphere Commerce 5.5 has added policy group subscription to the access control infrastructure.

In WebSphere Commerce 5.4, a policy was applied to resources owned by descendants of the policy owner. If different organizations in the same organization hierarchy wanted different levels of access control, achieving the different levels could be difficult. Furthermore, if the organization hierarchy was very deep, understanding all the policies that applied to an organization close to the bottom of the hierarchy could be confusing.

In order to make things simpler and more explicit in WebSphere Commerce 5.5, policies are first grouped into policy groups, based on business and access control requirements. For example, one policy group could have the policies needed to support contracts, while another could allow only registered users to shop. Then, depending on an organization's business and access control requirements, the organization would explicitly subscribe to the appropriate policy groups. When an organization subscribes to policy groups, only the policies in those policy groups will apply to the organization's resources. Its ancestor organizations' policies will not apply. However, if an organization does not explicitly subscribe to policy groups, it will inherit the policy subscription of its closest ancestor that is subscribing.

For an overview of policy groups, see the section discussing "Policy groups" in Chapter 3, "Authorization concepts," on page 17.

Security improvements in WebSphere Commerce 5.4

The following section lists the security enhancements in WebSphere Commerce 5.4 relative to WebSphere Commerce Suite 5.1 and retained in WebSphere Commerce 5.5. Most of these enhancements were made in the WebSphere Commerce Business Edition 5.1 release. These enhancements are generally applicable to the:

- WebSphere Commerce site administrator
- System administrator
- WebSphere Commerce developer

Note that sometimes these roles are interchangeable.

Enhancements for the Site Administrator

The following are WebSphere Commerce security enhancements that are generally targeted to a site administrator:

Access control

- **Access control framework —** A key enhancement is that a new access control framework has been implemented in WebSphere Commerce 5.4 and retained in WebSphere Commerce 5.5 (along with the new policy group enhancement in WebSphere Commerce 5.5). This new framework uses access control policies to determine if a given user is permitted to perform a given action on a given resource. The new access control framework provides fine-grained access control. It works in conjunction with, but does not replace the access control provided by the WebSphere Application Server. The new access control framework is described in detail in Part 3, "Administering security authorization," on page 83.

  The new access control framework enhances the previous access control in the following ways:

  It is expressive...
  It captures the intent of a large variety of access policies. The framework is generic so that it can handle a vast array of user groups, resource groups, actions groups and relationship groups.

  It is hierarchical...
  Access control policies belong to policy groups. Policy groups to which an organization subscribes can also be implicitly applied to its sub-organizations.

  It is customizable...
  Access control policies are externalized from the application code, so changes to policies can be made without recompiling code.

  It is compact...
  The new framework scales well. The number of access control policies grows with the number of business processes and not the number of objects. Most of the grouping framework is based on implicit conditions, so as long as the conditions are satisfied, the policy will apply.
- **Cross-site scripting —** Reject any user request that contain attributes or characters that are designated as not allowed, using the Cross Site Scripting Protection node of the WebSphere Commerce Configuration Manager. It is described in detail in Chapter 4, "Enhancing site security," on page 49.

Authentication

- **Password storage —** WebSphere Commerce encrypts and stores a one-way hash of passwords using the SHA-1 hashing scheme in the WebSphere Commerce database, rather than storing the passwords themselves. This ensures that user passwords are not decipherable by anyone, including the site or system administrator.
- **Password Invalidation —** Require users to change their passwords when they are logging in to the system for the first time, using the Password Invalidation node of the WebSphere Commerce Configuration Manager. It is described in detail in Chapter 4, "Enhancing site security," on page 49.
- **Account policy —** Set up an account policy for your site to define the account-related policies in use, by using the Account policy page of the WebSphere Commerce Administration Console. It is described in detail in Chapter 4, "Enhancing site security," on page 49.
- **Password policy —** Set up a password policy for your site to control a user's password selection characteristics using the Password policy page

of the WebSphere Commerce Administration Console. It is described in detail in Chapter 4, “Enhancing site security,” on page 49.

- **Account Lockout policy —** Set up an account lockout policy for your site to reduce the chances of a user account being compromised using the Account lockout policy page of the WebSphere Commerce Administration Console. It is described in detail in Chapter 4, “Enhancing site security,” on page 49.

Authorization

Password protected commands — Require users to enter their passwords if they are running requests that run designated commands, using the Password Protected Commands node of the WebSphere Commerce Configuration Manager. It is described in detail in Chapter 4, “Enhancing site security,” on page 49.

Encrypted data

Database update tool — Update encrypted data such as passwords and credit card information as well as the merchant key in a WebSphere Commerce database, using the Database Update Tool node of the WebSphere Commerce Configuration Manager. It is described in detail in Chapter 4, “Enhancing site security,” on page 49.

Session management

Login Timeout — Log off a user that is inactive for an extended period and request they log back on to the system, using the Login Timeout node. This enhancement is invoked through the WebSphere Commerce Configuration Manager and is described in detail in Chapter 4, “Enhancing site security,” on page 49.

Logging

Access logging — Quickly identify any security threats against WebSphere Commerce by enabling access logging. This enhancement is invoked through the WebSphere Commerce Configuration Manager and is described in detail in Chapter 4, “Enhancing site security,” on page 49.

Enhancements for the System Administrator

The following are security enhancements made in WebSphere Commerce 5.4 and retained in WebSphere Commerce 5.5 that are generally targeted to a site administrator:

- An important security enhancement is the ability to configure the WebSphere Commerce administrative tools to run on a nonstandard port number (for example, port 8000 as opposed to port 443). By restricting access to this port, you can limit access to the administration tools to your local network or intranet.
- From the WebSphere Commerce Administration Console Launch a security program that checks and deletes temporary WebSphere Commerce files that may contain potential security exposures using the Launch security check page.

Enhancements for the WebSphere Commerce Programmer

A key enhancement is that a new access control framework was implemented in WebSphere Commerce 5.4 and retained in WebSphere Commerce 5.5. This framework uses access control policies to determine if a given user is permitted to perform a given action on a given resource. The new access control framework provides fine-grained access control. It works in conjunction with, but does not replace the access control provided by the WebSphere Application Server. The new access control framework is described in detail in Part 3, “Administering security authorization,” on page 83.

The new access control framework enhances the previous access control in the following ways:

It is expressive...
: It captures the intent of a large variety of access policies. The framework is generic so that it can handle a vast array of user groups, resource groups, actions groups and relationship groups.

It is hierarchical...
: Access control policies owned by an organization are also applied to sub-organizations.

It is customizable...
: Access control policies are externalized from the application code, so changes to policies can be made without recompiling code.

It is compact...
: The new framework scales well. The number of access control policies grows with the number of business processes and not the number of objects. Most of the grouping framework is based on implicit conditions, so as long as the conditions are satisfied, the policy will apply.

For more information on security considerations for programmers, see the *WebSphere Commerce Programming Guide and Tutorials* document.

Security improvements in WebSphere Commerce Suite 5.1 Pro Edition

While Commerce Suite 5.1 represented a new e-commerce architecture and was a complete rewrite of the C++-based Commerce Suite 4.1, it contained all the security features of previous WebSphere Commerce Suite versions, plus it added new security improvements. These improvements have been inherited by WebSphere Commerce 5.5.

Commerce Suite 5.1 continued the protection against unauthorized access to WebSphere Commerce Suite administrators and shoppers resources that was provided by earlier releases by:

- Continuing support for access control features that ensure the WebSphere Commerce Suite user is either authenticated or in SSL mode before gaining access to or submitting sensitive information.
- Assigning WebSphere Commerce Suite commands to groups such that only the Site Administrator or Store level Administrators can execute a specific command, followed the same model as Commerce Suite 4.1.

General Security Enhancements

With the rewrite of Commerce Suite 5.1 in Java™, a number of inherent security problems that plagues software written in C++ were removed. Java does not use pointers, thus it has eliminated the buffer overflow problem that is a security vulnerability of most C++ based software. By complying with the industry standard J2EE specifications, WebSphere Commerce uses strong type checking to ensure the server does not execute rogue statements specified by devious individuals.

The industry standard Triple DES (data encryption standard) algorithm was used to protect sensitive information in the WebSphere Commerce system. The package containing the Triple DES algorithm is digitally signed such that if the package were tampered the WebSphere Commerce server would not start. These enhancements are retained in WebSphere Commerce 5.5.

Session Management

The WebSphere Commerce session management was completely rewritten for maximum security, using a unique technique to ensure cookies are not stolen. By using an authentication cookie that only flows over SSL (secure sockets layer) and consist of an encrypted timestamp, the rewritten session management design guarded against session hijacking.

Authentication

System and application passwords needed by the WebSphere Commerce server during execution were securely encrypted, using a merchant specified 128– bit key, and stored in the WebSphere Commerce configuration files. Sensitive information that appears in the users URL entry box is encrypted to protect shoppers from unauthorized disclosure.

Logging

The WebSphere Commerce log system was designed with security as a key consideration so that sensitive information such as shopper's password and credit card information was not logged by default to the WebSphere Commerce log files.

Chapter 2. Authentication

WebSphere Commerce views authentication as the process of verifying that users or applications are who they claim to be. This section describes the details of several aspects of WebSphere Commerce authentication.

WebSphere Commerce authentication model

The WebSphere Commerce authentication model is based on the following concepts:

- Challenge mechanisms
- Authentication mechanisms
- User registry

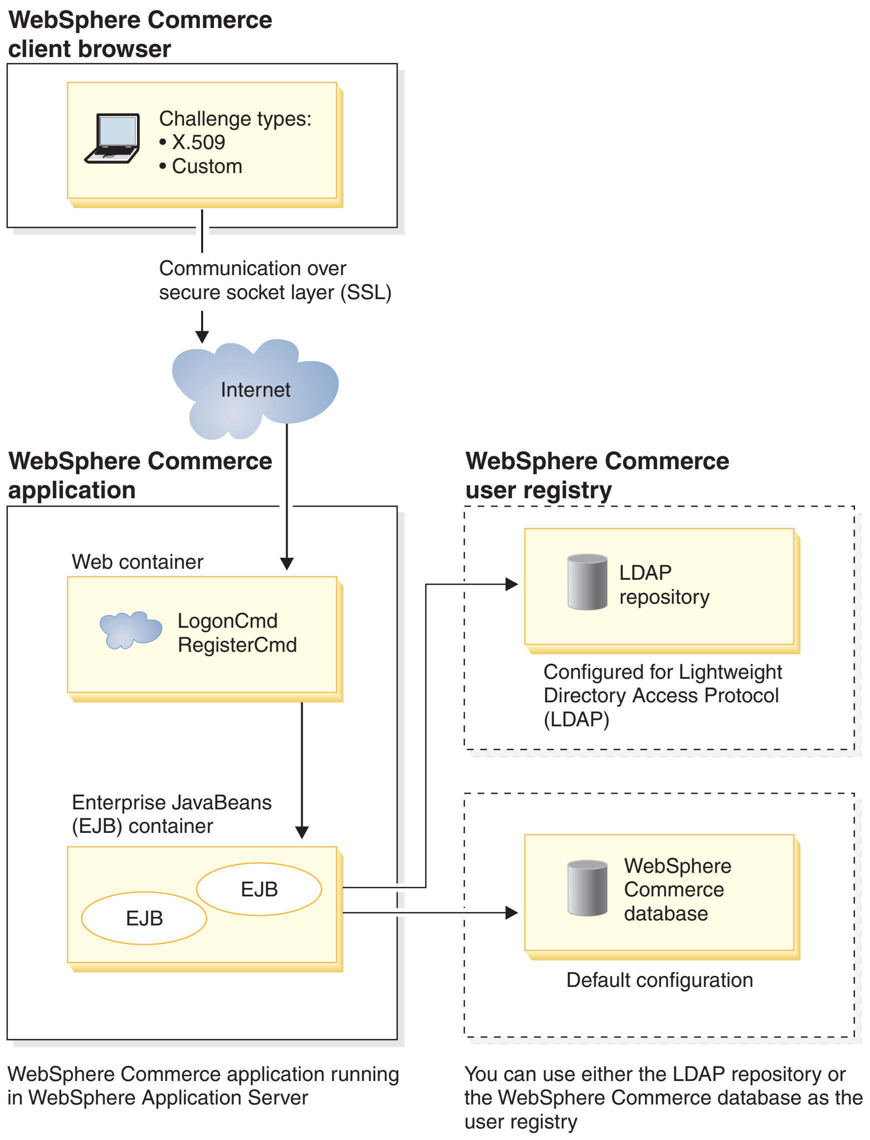

WebSphere Commerce client browser
Challenge types:
• X.509
• Custom
Communication over secure socket layer (SSL)
Internet
WebSphere Commerce application
Web container
LogonCmd
RegisterCmd
Enterprise JavaBeans (EJB) container
EJB
EJB
WebSphere Commerce user registry
LDAP repository
Configured for Lightweight Directory Access Protocol (LDAP)
WebSphere Commerce database
Default configuration
WebSphere Commerce application running in WebSphere Application Server
You can use either the LDAP repository or the WebSphere Commerce database as the user registry

Figure 1. WebSphere Commerce security model

Challenge mechanisms

A challenge mechanism specifies how a server challenges and retrieves authentication data from a user. WebSphere Commerce supports the following authentication methods or challenge mechanisms:

Form-based or custom authentication
This authentication mechanism permits a site or store specific login through an HTML page or a JSP form.

Certificate-based authentication (X.509 certificate)
The certificate challenge mechanism implies that the Web server is configured to perform mutual authentication over SSL. The client is required to present a certificate in order to establish the connection. This certificate is then credential mapped to a user registry.

Authentication mechanisms

An *authentication mechanism* authenticates a user by verifying the user's authentication data against an associated user registry. WebSphere Commerce issues an authentication token that is associated with a user on every subsequent request after the authentication process. It is terminated when the user logs off or closes the browser.

Certificate validation

This is the process of verifying that the X.509 client certificate is trusted by the Web server and that it complies with the Web server's certificate policy. WebSphere Commerce also verifies the X.509 certificate against the WebSphere Commerce database. The Web server performs the coarse-grain access control on the certificate, while WebSphere Commerce performs a fine-grain access control on the certificate.

LDAP bind

This is process of verifying the challenge information supplied is valid, by performing an LDAP bind operation to authenticate the user.

Database bind

This is the process of verifying the user ID and password supplied during the authentication process is valid when compared to the authentication information stored in the WebSphere Commerce database.

User registry

The user registry is a repository that contains user information, and the user's authentication information (for example, the password). Authentication information provided by a principal (that is, the representation of a human user or system entity in a user registry) can be verified or validated against the user registry.

WebSphere Commerce supports user registries based on two user domains: LDAP user registry and the WebSphere Commerce database.

WebSphere Commerce supports the following LDAP providers:

- AIX 400 Linux Solaris Windows IBM SecureWay® Directory
- AIX Solaris Windows Netscape Directory Server
- 2000 Windows 2000 Active Directory

Credentials

The WebSphere Commerce server supports authentication mechanisms based on validating credentials, such as certificates, tokens, or user ID and password pairs. Credentials are verified against a user registry that supports such a scheme.

WebSphere Commerce token

WebSphere Commerce uses a secure authentication cookie to manage authentication data. An authentication cookie flows only over SSL, and is time-stamped for maximum security. This cookie is used to authenticate the user under SSL-connections whenever a sensitive command is executed, for example, the `DoPaymentCmd`, which asks for a users credit card number. There is minimal risk that this cookie could be stolen and used by an unauthorized user.

A second cookie that flows between the browser and server under either SSL or non-SSL connection is used for verification of the user under non-SSL connections.

WebSphere Application Server LTPA token

An LTPA token is a piece of data that contains user information necessary to determine access permissions for a resource that is requested by the user. It contains the authentication data along with the digital signature of the WebSphere Application Server LTPA server.

In the case of the WebSphere Application Server Lightweight Third Party Authentication scheme, an LDAP directory containing the information about the users is the user registry against which authentication is performed. The resource server contacts the WebSphere Application Server Security Server and specifies LTPA to be the authentication mechanism. It also supplies the authentication data associated with the request. The WebSphere Application Server Security Server then validates the authentication data against the LTPA server and returns an LTPA token.

Single sign-on

The philosophy behind the HTTP single sign-on is to preserve user authentication to different Web Applications. Its goal is to: avoid prompting the user multiple times for security credentials within a given trust domain that includes:
- Cooperating but disparate WebSphere Application Server servers.
- Cooperating applications such as LDAP servers such as IBM SecureWay Directory Server.

In a single sign-on (SSO) scenario, an HTTP Cookie is used to propagate a user's authentication information to disparate Web servers relieving the user from entering authentication information for every new client-server session (assuming basic authentication).

For the steps to implement single sign-on with WebSphere Commerce, see Chapter 7, "Single sign-on," on page 77.

Authentication policies

An authentication policy is a set of rules that are applied to the authentication process and to the verification of authentication data by WebSphere Commerce. WebSphere Commerce supports account policies, other authentication-related policies, and session policies as described in the following sections.

Account policies

The following sections describe account policies available with WebSphere Commerce:

Account policy

The Account policy page of the WebSphere Commerce Administration Console allows you to set up an account policy. An account policy defines the account-related policies such as password and account lockout policies.

Once you have created an account policy, you can assign the policy to a user. Note that you cannot delete an account policy if it is in use (that is, a user is assigned the account policy).

For information on creating account policies, see “Setting up an account policy” on page 59.

Also see the reference topic ″Default Authentication Policies″ in the WebSphere Commerce online help.

Account lockout policy

The Account lockout policy page of the WebSphere Commerce Administration Console allows you to set up an account lockout policy for different user roles within WebSphere Commerce. The account lockout policy disables a user account if malicious actions are launched against that account in order to reduce the chances that the actions compromise the account.

The account lockout policy enforces the following items:

- The account lockout threshold. This is the number of invalid logon attempts before the account is disabled.
- Consecutive unsuccessful login delay. This is the time period for which the user is not allowed to login, after two failed attempts to login. The delay gets incremented by the configured time delay value (for example, 10 seconds) with every consecutive login failure.

For information on creating account lockout policies, see “Setting up an account lockout policy” on page 61.

Password policy

The Password policy page of the WebSphere Commerce Administration Console allows you to control a user’s password selection in order to define the characteristics of the password to ensure that it complies with the security policy for your site.

This feature defines attributes with which the password must comply. The password policy enforces the following conditions:

- Whether the user ID and password can match.
- Maximum occurrence of consecutive characters.
- Maximum instances of any character.
- Maximum lifetime of the passwords.
- Minimum number of alphabetic characters.
- Minimum number of numeric characters.
- Minimum length of password.
- Whether the user’s previous password can be reused.

For information on creating password policies, see “Setting up a password policy” on page 59.

Also see the reference topic ″Default Authentication Policies″ in the WebSphere Commerce online help.

Other authentication-related policies

The following sections describe the other authentication-related policies available with WebSphere Commerce:

Password Invalidation

Use the Password Invalidation node of the Configuration Manager to enable or disable the password invalidation feature. This feature, when enabled, requires WebSphere Commerce users to change their password if the user's password has expired. In that case, the user is redirected to a page where they are required to change their password. Users are not able to access any secure pages on the site until they have changed their password.

For information on using the Password Invalidation node, see "Activating password invalidation" on page 53.

Password Protected Commands

Use the Password Protected Commands node of the Configuration Manager to enable or disable the password protected commands feature. When this feature is enabled, WebSphere Commerce requires registered who are logged onto WebSphere Commerce to enter their password before continuing a request that runs designated WebSphere Commerce commands.

Caution: When you configure the password protected commands, some of the commands shown in the command selection list can be executed by generic or guest users. Configuring such commands as password protected will restrict generic and guest users from running them. Therefore, you should exercise caution when you configure commands to be password protected.

Note: WebSphere Commerce will only display the commands that are designated as `authenticated` or set with the `https` flag in the `URLREG` table in the list of available commands.

For information on using the Password Protected Commands node, see "Enabling password protected commands" on page 54.

Session policies

In WebSphere Commerce, session policies are embodied in the login timeout policy.

With the login timeout policy, WebSphere Commerce will log off a user that is inactive for an extended period and request they log back on to the system using the Login Timeout node. This enhancement is invoked through the WebSphere Commerce Configuration Manager and is described in detail in "Enabling login timeout" on page 53.

Chapter 3. Authorization concepts

WebSphere Commerce views access control or authorization as the process of verifying that users or applications have sufficient authority to access a resource. This section describes the details of several aspects of WebSphere Commerce access control.

Authorization or access control, in WebSphere Commerce is accomplished using access control policies. An access control policy is a rule that describes which group of users can perform a set of actions on a set of resources. WebSphere Commerce provides a set of default access control policies. These default access control policies are specified in XML format and are designed to address many of the typical access control requirements that an e-commerce site needs.

Business models

In WebSphere Commerce 5.4, after you created your instance, the Site Administrator had to make the following decisions:

1. The organizational structure appropriate for the site
2. The roles to be assigned to particular organizations
3. The access control policies that would be needed

After all of these decisions were made, the store could then be published against the appropriate organization.

In WebSphere Commerce 5.5, this process has been simplified by the creation of business models. A business model provides the organization structure, roles, access control polices, and predefined stores that are targeted at a specific e-commerce solution. Business models can be used as the development stage as a base, to which content can be added, deleted, or changed.

The following business models are available with WebSphere Commerce 5.5:

- Consumer direct
- B2B direct
- Demand chain
- Hosting
- Supply chain

In order to understand the business models, and the access control component of WebSphere Commerce, you must first understand the typical organizational hierarchy of an e-commerce site.

Note: For more information on the business models, see the *WebSphere Commerce Fundamentals Guide*.

Organizational hierarchy

Users and organizational entities within the WebSphere Commerce member subsystem are organized into a hierarchy. This hierarchy emulates a typical organizational hierarchy, with entries for organizations and organizational units, and entries for users in the leaf nodes. The hierarchy includes an artificial organizational entity called a *root organization* at the top. All other organizational entities and users are descendants of this root organization. Under the root organization there can be one seller organization and several buyer organizations; all these organizations can have one or more sub-organizations under them. Buyer or Seller Administrators are the heads of the organizations, and they are responsible for maintaining their organizations. On the seller organization side, each sub-organization can have one or more stores within it. Store Administrators are responsible for maintaining the stores. The following diagram shows the organizational hierarchy of a business-to-business e-commerce site.

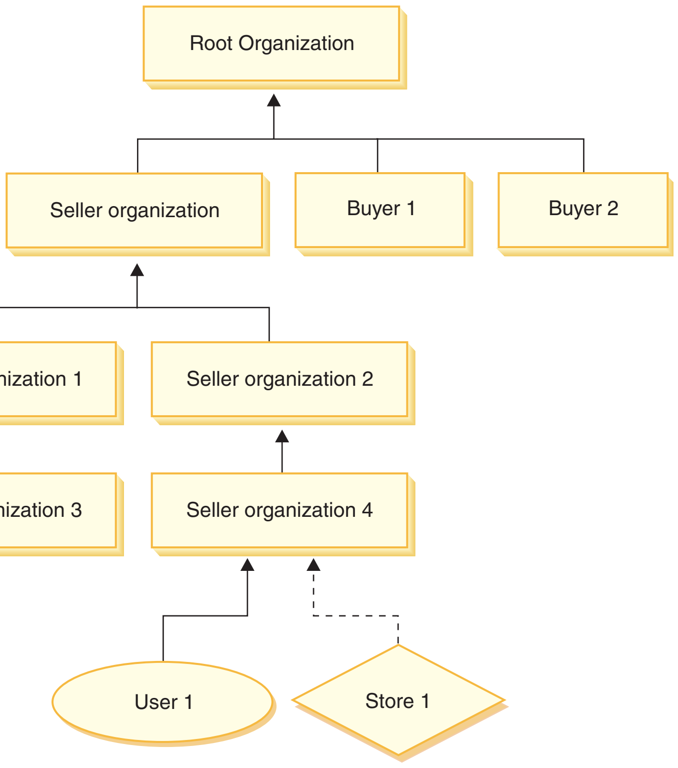


Figure 2. Organizational hierarchy of a business-to-business site

Root organization

The root organization is at the top of the organizational hierarchy. A Site Administrator has super user access to perform any operation within WebSphere Commerce. The Site Administrator installs, configures, and maintains WebSphere Commerce and its associated software and hardware. This role typically controls access and authorization (that is, creating and assigning members to the appropriate role) and manages the Web site. The Site Administrator can assign roles to users and specify the organization(s) for which the user plays the role. The Site Administrator must assign a password to each administrator to ensure that only authorized parties can access confidential information. This provides a way to control key responsibilities, such as updating a catalog or approving a request for quotation (RFQ).

Note: It is possible for a user to play roles in an organization other than their parent organization.

In a WebSphere Commerce site, there is one seller organization. In a business-to-business site, there are also one or more buyer organizations. The Site Administrator may define both the access control policies of the seller organization (that owns the store) as well as the access control policies of each organization that buys from the store. In a business-to-consumer site, there are no buyer organizations. Business-to-consumer customers are modeled as members of the default organization.

Organizations (seller)

Both in business-to-business and business-to-consumer sites, the Site Administrator creates one top-level seller. Underneath this seller organization, other sub-organizations or organization units can be created. Any of these sell-side organizational entities can own one or more stores. The Site Administrator then defines any special access control policies for a seller organization, and assigns a Seller Administrator to manage that organization. The Seller Administrator registers users, assigns them different roles to fit the organization's business needs according to the access control policies pertaining to that organization.

The Seller Administrator's responsibilities are summarized as follows:

- Create sub-organizations that can own stores. Optionally, define which processes within the organization require approval. This step is only required in a business-to-business site.
- Assign roles to the sub-organizations.
- Create users.
- Assign roles to users.

Organizations (buyer)

In a business-to-business site, the Site Administrator creates one or more buyer organizations, depending on the business needs. The Site Administrator then defines any special access control policies for a buyer organization and assigns a Buyer Administrator to manage the buyer organization. The Buyer Administrator registers users and assigns them different roles to fit the organization's business needs, according to the access control policies pertaining to that organization.

The Buyer Administrator's responsibilities are summarized as follows:

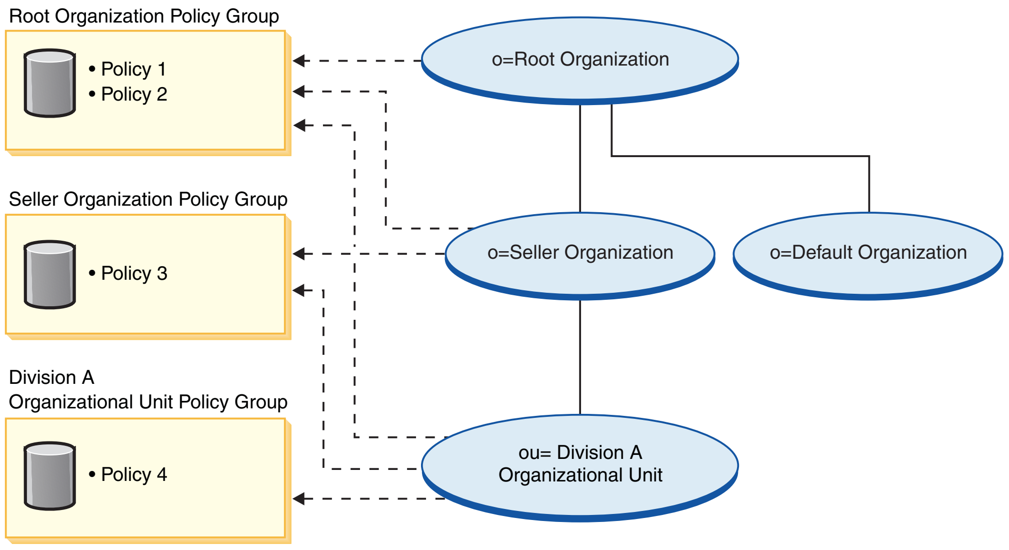
- Create and administer the sub-organizations within the buyer organization. Optionally, define which processes within the organization require approval. This step is only required in a business-to-business site.
- Assign roles to the sub-organizations.
- Create users.
- Assign roles to users.

Note: The Site Administrator can modify and manage the access control policies of the buyer organization if appropriate. For more information on the Site Administrator's tasks, see the WebSphere Commerce online help.

Policy groups

WebSphere Commerce 5.5 supports various business models, and each business model has it's own set of access control policies. In order to group the sets of policies within the models, policy groups were created. Policies are explicitly assigned to appropriate policy groups and then organizations can subscribe to one or more of these policy groups. For example, in the following diagram, Seller Organization subscribes to Seller Organization Policy Group, and Root

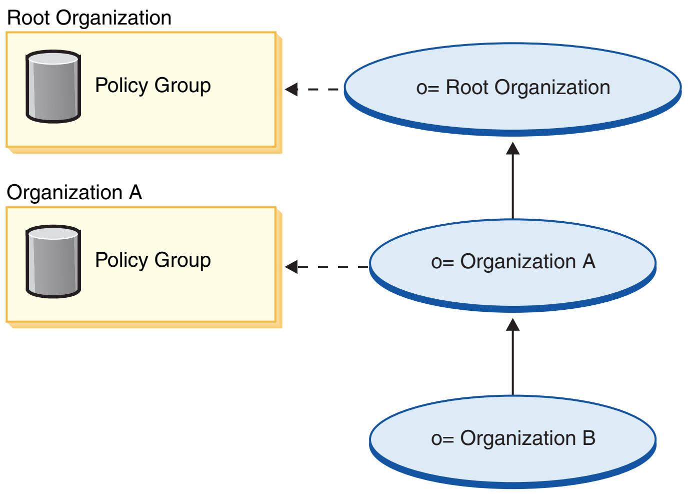
Organization Policy Group.



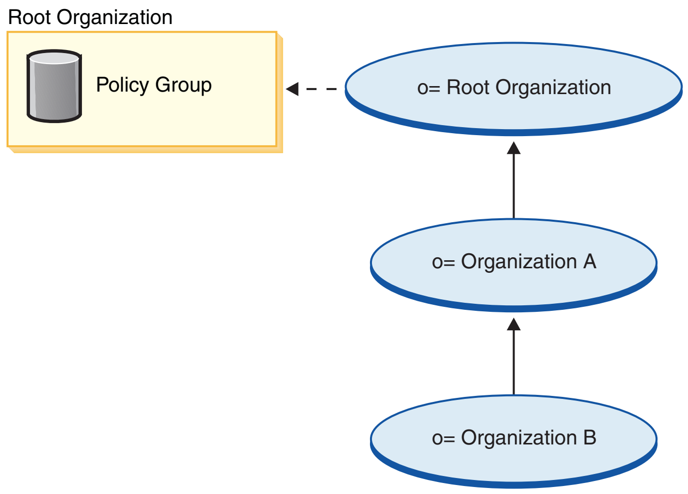
Policies are assigned to policy groups. For example, in the preceding diagram, Policy 1 and Policy 2 are assigned to the Root Organization Policy group, Policy 3 is assigned to the Seller Organization Policy Group, and Policy 4 is assigned to the Division A Organizational Unit Policy Group.

Policy group subscription

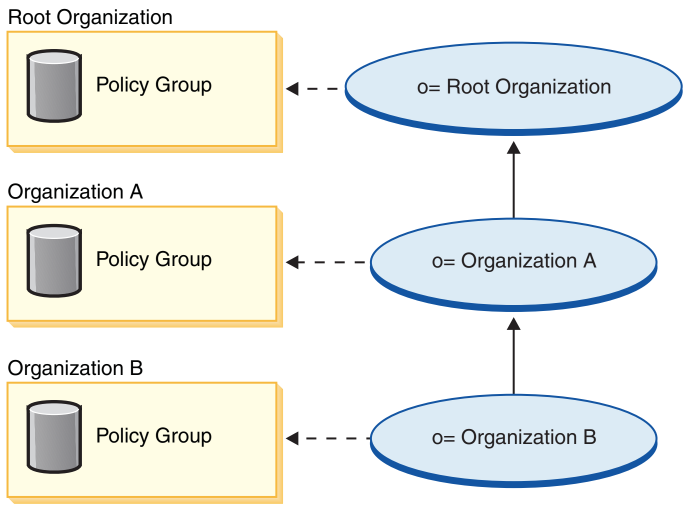
In previous versions of WebSphere Commerce, a policy applied to all resources owned by the descendants of that policy's owner organization. For example, if Organization A had a certain policy and was the parent of Organization B, then Organization B implicitly, had that policy as well. In WebSphere Commerce 5.5, organizations can now subscribe to policy groups. In WebSphere Commerce 5.5 , if Organization B does not subscribe to any policy groups, the access control framework will begin searching up the organization hierarchy until it encounters an organization that subscribes to at least one policy group. If Organization B's immediate parent organization, Organization A, subscribes to a policy group, the searching stops, and the policies are applied to Organization A and B. This can be seen in the following diagram.



If Organization A does not subscribe to a policy group, the search continues up the organization hierarchy, until an organization with a subscription is reached. This is seen in the following diagram where the Root Organization subscribes to a policy group. The policies in that group apply to Organization B and Organization A.



If Organization B subscribes to a policy group, the search stops at Organization B. So only the policies in the Organization B policy group will apply to Organization B.



Access control policy

An access control policy authorizes a group of users to perform a set of actions on a set of resources within WebSphere Commerce. Unless authorized through one or more access control policies, users have no access to any functions of the system. To understand access control policies you need to understand four main concepts: users, actions, resources, and relationships. Users are the people who use the system. Resources are objects in the system that need to be protected. Actions are the activities that users can perform on the resources. Relationships are optional conditions that exist between users and resources.

Elements of an access control policy

An access control policy consists of four elements:

Access group
The group of users to which the policy applies.

Action Group
A group of actions performed by the user on resources.

Resource group
The resources controlled by the policy. A resource group may include business objects like `contract` or `order`, or a set of related commands such as, all the commands that users of a particular role can perform.

Relationship (optional)
Each resource class can have a set of relationships associated with it. Each resource can have a set of users that fulfill each relationship. For example, a policy could specify that only the creator of an order can modify it. In this case, the relationship would be `creator` , and it is between the user and the order resource.

Access control policy concepts

Access control policies grant users access to your site. Unless they are authorized to perform their responsibilities through one or more access control policies, users have no access to any of your site's functions.

Each access control policy takes the following form:

```
AccessControlPolicy [AccessGroup,ActionGroup,ResourceGroup,Relationship]
```

The elements in the access control policy specify that a user belonging to a specific access group is permitted to perform actions in the specified action group on resources belonging to the specified resource group, as long as the user satisfies a particular relationship with respect to the resource. The relationship is only specified when needed. For example, `[AllUsers,UpdateDoc,doc,creator]` specifies that all users can update a document, if they are the creator of the document.

The following sections describe conceptual information and terminology associated with access control.

Member groups

The Member subsystem in WebSphere Commerce allows you to create member groups, which are groups of users categorized for various business reasons. The groupings can be used for many purposes, for example, access control purposes, approval purposes, as well as for marketing purposes such as calculating discounts and prices, and displaying products. A member group of type Access Group (-2) is for access control purposes, while a member group of type User Group (-1) is for general use. A member group is associated with member group types in the `MBRGRPUSG` table.

Access groups: A member group of type Access Group (-2) is for grouping users for access control purposes. An access group is one element of an access control policy. The criteria for membership in an access group is usually based on roles, the organization to which the user belongs, or the user's registration status. For example, the access group called `Buyer Administrators` is a group whose users play the role of Buyer Administrator.

WebSphere Commerce includes a number of default roles, and corresponding to each role is a default access group that implicitly references that role. Roles can be used as attributes to add users to an access group based on the type of activities they perform in the site. For example, by default there is a role called Seller Administrator and a corresponding access group called `Seller Administrators`. A Site Administrator uses the WebSphere Commerce Administration Console to create, maintain, and delete access groups for a site. A Site Administrator, Buyer Administrator, Seller Administrator, or Channel Manager uses the WebSphere Commerce Organization Administration Console to assign roles to users or to explicitly assign users to access groups.

Implicit access group: An implicit access group is defined by a set of criteria. Anyone who satisfies the criteria is a member of the group. The criteria are usually based on a user's roles, parent organization, or registration status. The implicit conditions that define membership in a member group are in the `CONDITIONS` column of the `MBRGRPCOND` table. Using implicit access groups that specify the attributes of users, makes it easy to authorize access to similar users without having to explicitly assign and unassign individual users. It also eliminates the need to update the members of a group when a user's attributes change. Furthermore, since multiple access groups can refer to the same user attribute, assigning an attribute to a user can implicitly include that user in multiple access groups. A simple criterion for an access group is to include everyone that has been assigned a specific role, regardless for which organization the user plays the role. A more complex criterion would be to specify that only users that play one of a possible set of roles for a particular organization would belong to the access group.

Explicit access group: It is possible to explicitly add or remove a user from a member group. Both of these explicit specifications can be done using the `MBRGRPMBR` table. An explicit access group contains explicitly assigned users who may or may not share common attributes. This also allows you to exclude individuals that satisfy the conditions for inclusion in an implicitly defined group, but that you want excluded anyway.

User groups: A member group of type User Group (-1) is a collection of users defined by the merchant, who share a common interest. User groups are similar to clubs that are offered by large stores for their frequent or preferred customers. Being part of a user group can entitle customers to discounts or other bonuses for purchasing products. For example, if market research shows that senior customers repeatedly purchase travel books and luggage, you can assign these customers to a member group called `Seniors' Travel Club`. Likewise, you can create a user group to reward frequent customers for their business.

Actions

Generally, an action is an operation that is performed on a resource. In role-based policies for controller commands, the action is `Execute` and the resource is the command being executed. In role-based policies for Views, the action is the name of the view, and the resource is `com.ibm.commerce.commands.ViewCommand`. For resource-level access control, actions typically map to WebSphere Commerce commands, and the resource is usually the remote interface of a protected EJB (Enterprise Java Bean). For example, the controller command `com.ibm.commerce.order.commands.OrderCancelCmd` operates on the `com.ibm.commerce.order.objects.Order` resource. Lastly, in databean policies, the `Display` action is used to allow the activation of databean resources.

The WebSphere Commerce Administration Console can be used by a Site Administrator to associate existing actions with action groups, but not for creating new actions. New actions can be created by defining them in an XML file and then loading them to the database. Actions are stored in the `ACACTION` table.

Action groups

Action groups are groups of related actions. An example of an action group is the `AccountManage` group that includes the following commands:

- `com.ibm.commerce.account.commands.AccountDeleteCmd`
- `com.ibm.commerce.account.commands.AccountSaveCmd`

Only the Site Administrator can create, update, and delete action groups. This can be done from the WebSphere Commerce Administration Console and through XML. Action groups are stored in the `ACACTGRP` table. Actions are associated with action groups in the `ACACTACTGP` table.

Resource category

Resource category refers to a class of resources that need to be protected by access control. Resources must implement the Protectable interface information. Resource categories are Java classes such as order, RFQ, and auction. Resources are the instances of these classes. For example, Auction1 created by Auction Administrator A is one resource; Auction2 created by Auction Administrator B is another resource. These two resources belong to the resource category: auction.

Note: For more information on the Protectable interface, see the *IBM WebSphere Commerce Programmer's Guide*.

Resource categories are defined in the `ACRESCGRY` table, and for convenience, are sometimes referred to as resources. A Site Administrator can associate existing resource categories with resource groups, using the WebSphere Commerce Administration Console. New resource categories can be created using XML.

Resources

Resources are any objects in the system that need to be protected. For example, RFQs, auctions, users, and orders are some of the resources in WebSphere Commerce which need to be protected. Each resource has an owner. The ownership of the resource is used to determine which access control policies apply to it. Access control policies have an owner, which is an organizational entity. A policy is only applied to resources that are owned by the same organizational entity that subscribes to a policy group, that contains the policy. If the organization that owns the resource does not subscribe to policy groups, then the policies in the policy groups that are subscribed by the closest ancestor organization will be applied.

Controller command resources: For role-based access control for controller commands, the policy is structured such that the `Execute` action is being performed on the controller command resource. These policies are intended to restrict the execution of controller commands to users with a specified role. The access group for these policies is usually those with a single role, for example, Product Managers (those with the Product Manager role). Then, the resource group would be the set of controller commands that a product manager can execute.

While enforcing role-based access control on a controller command, the owner of the command must be determined. This is done by calling the `getOwner()` method on the command if it has been implemented. Usually this method is not implemented, so WebSphere Commerce Runtime will evaluate it by doing one of the following:

- Use the organization that owns the store that is currently in the command context.
- If there is no store in the command context, use the Root Organization as the owner.

Data bean resources: Not all data beans require protection. Within the existing WebSphere Commerce application, data beans that require protection already implement the required access control. The question of what to protect comes into play when you create new data beans. Deciding which resources to protect depends upon your application. A data bean should be protected (directly or indirectly), if the information to be displayed is not sufficiently protected by the role-based access control on the view, that corresponds to the JSP (Java Server Page) that contains the data bean.

If a data bean needs to be protected and can exist on its own, it should be directly protected. If the existence of a data bean depends upon the existence of another data bean, then it should delegate to the other data bean for protection. An example of a data bean that would be directly protected is the `Order` data bean. An example of a data bean that would be indirectly protected is the `OrderItem` data bean, as it cannot exist without `Order` data bean. Refer to the *WebSphere Commerce Programming Guide and Tutorials* for more information on how to protect the data bean resource.

Data resources: Data resources refer to business objects that can be manipulated such as, auctions, orders, RFQs, and users. These are usually protected at the enterprise bean level, but it is possible to protect any class, as long as it implements the Protectable interface. Data resources are protected using resource-level access control checks. The common way of doing this by returning data resources in the `getResources()` method of a controller or task command. For more information see the *WebSphere Commerce 5.4 Programmer's Guide* .

Resource groups

A resource group identifies a set of related resources. A resource group can include business objects such as a contract or a set of related commands. In access control, resource groups specify the resources to which the access control policy authorizes access.

Resource groups are defined in the `ACRESGRP` table. Site Administrators can manage resource groups and associate resources with resource groups using the WebSphere Commerce Administration Console, or by using XML.

Implicit resource groups: Implicit resource groups define resources that match a certain set of attributes. One of these attributes must be the Java class name. Other attributes may include status, store ID, price, etc. For example, you could create an implicit resource group that includes all orders that have pending status (`ORDERS.STATUS=P`). Implicit resource groups are usually used for grouping resources that will be used in resource-level policies, when the resources share a common attribute beyond the Java class name.

Implicit resource groups are defined using the `CONDITIONS` column of the `ACRESGRP` table. Simple implicit resource groups can be created using the WebSphere Commerce Administration Console. Increasingly complex groups can be created using XML.

Explicit resource groups: Explicit resource groups are specified by associating one or more resource categories to a resource group. This association is done in the `ACRESGPRES` table. Adding a resource category to a group explicitly, by listing its Java class name, lets you group individual resources that might not necessarily share common attributes.

Relationships

Each resource may have some kind of relationship associated with it, and a set of members that fulfill each relationship. For example, all resources have a relationship of *owner*, which is fulfilled by the owner of the resource. Other relationships can include recipients of documents and the creator of an order. These resource relationships are important in determining who can perform certain actions on a particular instance of a resource. For example, the creator of a document may not be able to delete it, but perhaps an auditor may. Similarly, a reviewer may only be able to read and approve a document, but not forward it or perform other operations.

Relationships are stored in the `ACRELATION` table, and are optionally specified in an access control policy, using the `ACRELATION_ID` column of the `ACPOLICY` table. When evaluating a policy that requires the fulfillment of a relationship between the user and the resource, the fulfills (`Long Member, String relationship`) method on the resource will be called to evaluate it. When comparing these relationships to relationship groups, these relationships are sometimes referred to as simple relationships.

Relationship groups: Access control policies can specify that a user must fulfill a particular relationship with respect to the resource being accessed, or they can specify that a user must fulfill the conditions specified in a relationship group. In most cases, a relationship is sufficient. However, if more complex relationships are needed, a relationship group can be used instead. A relationship group allows you to specify multiple relationships and also a chain of relationships. Both of these are done using a relationship chain construct. A relationship chain is a construct that can express a simple relationship (directly between a user and the resource), but can also be used to express a series of relationships between the user and the resource. For example, in order to express that a user must have a role in an organization that has a relationship (other than the owner relationship) with the resource, one must use a relationship group. In this example, there is a role relationship between the user and the organization, and a relationship between the organization and the resource.

Comparing relationships and relationship groups: In most cases, using a relationship should satisfy the access control requirements for your application since, conceptually, most relationships are directly between a user and the resource. For example, the policy states that the user must be the creator of the resource. If however, you need to specify multiple relationships, a relationship group should be used. For example, the policy states that the user must be the creator or the submitter of the resource.

Relationship groups are also needed to express a chain of relationships between a user and the resource. In a chain of relationships, there is no direct relationship between the user and the resource for example, a user belongs to the buying organization specified by an order. In this case, the user has a child relationship with the organization, and that organization has a buying relationship with the order.

Relationship chains: Each relationship group consists of one or more `RELATIONSHIP_CHAIN` open conditions, grouped by `andListCondition` or `orListCondition` elements. A relationship chain is a series of one or more relationships. The length of a relationship chain is determined by the number of relationships it consists of. This can be determined by examining the number of `<parameter name= "X" value="Y"/>` entries in the XML representation of the relationship chain. The following is an example of a relationship chain with a length of one.

```
<openCondition name="RELATIONSHIP_CHAIN">
<parameter name="RELATIONSHIP"
value="aValue"/>
</openCondition>
```

For relationship chains of length one, the `<parameter name="Relationship" value="something">` element specifies a direct relationship between the user and the resource. The value attribute is the string representing the relationship between the user and the resource. It must also correspond to the relationship parameter of the `fulfills()` method on the protectable resource.

When a relationship chain has a length of two, it is a series of two relationships. The first ,`<parameter name= "X" value="Y"/>`, element is between a user and an organizational entity. The last, `<parmeter name= "X" value="Y"/>`, element is between that organizational entity and the resource. The following is an example of a relationship chain with a length of two.

```
<openCondition name=RELATIONSHIP_CHAIN">
<parameter name="aValue1" value="aValue2"/>
<parameter name="RELATIONSHIP" value="aValue3"/>
</openCondition>
```

The `aValue1` possible values include `HIERARCHY` and `ROLE`. `HIERARCHY` specifies that there is a hierarchical relationship between the user and the organizational entity in the membership hierarchy. `ROLE` specifies that the user plays a role in the organizational entity.

If the value of `aValue1` is `HIERARCHY`, the possible values include `child`, which returns the organizational entity for which the user is a direct child in the member hierarchy. If the value of `aValue1` is `ROLE`, possible values include any valid entries in the NAME column of the ROLE table which return all of the organizational entities for which the current user plays this role.

The `aValue3` entry, is a string representing the relationship between one or more organizational entities retrieved from evaluating the first parameter and the resource. This value corresponds to the relationship parameter of the `fulfills()` method on the protectable resource. If more than one organizational entity was returned by evaluating parameter `aValue1` , this part of the `RELATIONSHIP_CHAIN` is satisfied if at least one of these organizational entities satisfies the relationship specified by parameter `aValue2`.

Note: A relationship group that consists of a single relationship chain with a single parameter element, is functionally equivalent to a simple relationship. In this case, it is easier to use relationship instead of relationship group in the policy. For more information on defining relationship groups, see "Defining relationship groups" on page 146.

Types of access control policies

There are two types of access control policies:

- Groupable standard policies (policy type -2)
- Groupable template policies (policy type -3)

Both groupable template and groupable standard policies must belong to a policy group in order to be applied in the system. A groupable standard policy is applied, once, at organizations that subscribe to a policy group that contains the policy.

Groupable template policies are dynamic in nature in that they have an access group that is scoped, when the system is running, to the organization that owns the resource. For example, when this type of policy is applied to a resource owned by Organization XYZ, it would check if the user played one of the specified roles for Organization XYZ or its ancestors.

Special default access control policies

The following policies require some extra explanation:

- Site Administrators Can Do Everything (SiteAdministratorsCanDoEverything)
- BecomeUser Customer Service Group Executes Become User Commands on customer's behalf (BecomeUserCustomerServiceGroupExecutesBecomeUserCmdsResourceGroup)

The SiteAdministratorsCanDoEverything policy, is a special default policy that grants super-user access to administrators with the Site Administrator role. In this policy, a Site Administrator can perform any action on any resource, even if those actions or resources have not been defined. It is important to be aware of this when assigning this role to users.

BecomeUserCustomerServiceGroupExecutesBecomeUserCmdsResourceGroup policy is a special policy that allows certain administrative users to run specified commands on behalf of other users. This policy is needed, for example, when a customer requests a customer service representative to create an order on his behalf. In this case, the customer service representative is able to run the command such that it appears like the customer himself has run the command.

Roles

As mentioned above, WebSphere Commerce provides default sets of roles. The Site Administrator must assign specific roles to every organization before assigning users to those roles. An organization can only take on roles that have been assigned to it's parent organization.

All roles in WebSphere Commerce are scoped to an organization. For example, a user plays the Product Manager role for Organization X. In this case, Organization X must support the Product Manager role. In general, an organization must support a role before any user can be assigned that role for that organization. The access control policies could then be setup such that this user can only perform product management operations within the context of Organization X and its sub-organizations.

Note: Assigning roles to users and organizations is done in the `MBRROLE` table.

The default roles that come with WebSphere Commerce can be grouped into the following categories:

- Technical operations roles
- Marketing roles
- Operational roles
- Customer service roles
- Business relationship roles
- Product management and merchandising roles

In WebSphere Commerce 5.5, each role is associated with one or more business models. Within, each model, a role can perform a select number of tasks using the Commerce Accelerator, Administration Console, and Organization Administration Console tools. For more information on the business models, see the *WebSphere Commerce Fundamentals*.

The following chart displays the access that each role has to each of the tools. Before assigning roles to users, ensure that you have the correct information on what access restrictions are applicable to that role.

Roles mapped to WebSphere Commerce tools for every store sample

Table 1. Roles mapped to WebSphere Commerce tools

Roles	**Samples**	**Tools**
Account Representative	• B2B direct: ToolTech	• Accelerator
Buyer Administrator	• B2B direct: ToolTech	• Organization Administration Console
Buyer Approver	• B2B direct: ToolTech	• Organization Administration Console
Buyer (sell-side)	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator
Buyer (buy-side)	• B2B direct: ToolTech • Hosting: Hosted store • Supply chain: Supplier hosted store	This role is available in the samples, but does not have access to any specific tool.
Category Manager	• Consumer direct: FashionFlow • B2B direct: ToolTech • Demand chain: Hosted store, Catalog asset store, • Hosting: Hosted store, Catalog asset store • Supply chain: Catalog asset store, Supplier hosted store	• Accelerator
Channel Manager	• Demand chain: Channel hub • Hosting: Hosting hub • Supply chain: Store directory	• Accelerator • Organization Administration Console
Customer Service Representative	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator

Table 1. Roles mapped to WebSphere Commerce tools (continued)

Roles	Samples	Tools
Customer Service Supervisor	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator
Logistics Manager	• B2B direct: ToolTech • Supply chain: Supplier hosted store	• Accelerator
Marketing Manager	• Consumer direct: Fashion Flow • B2B direct: ToolTech • Demand chain: Channel hub, Hosted store, Reseller storefront asset store • Hosting: Hosted store, Hosted storefront asset store	• Accelerator
Operations Manager	• Consumer direct: Fashion Flow • Demand chain: Hosted store • Hosting: Hosted store	• Accelerator
Pick Packer	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator
Procurement Buyer	• B2B direct: ToolTech • Supply chain: Supplier hosted store	This role is available in the samples, but does not have access to any specific tool.
Procurement Buyer Administrator	• B2B direct: ToolTech • Supply chain: Supplier hosted store	This role is available in the samples, but does not have access to any specific tool.
Product Manager	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator
Receiver	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator

Table 1. Roles mapped to WebSphere Commerce tools (continued)

Roles	Samples	Tools
Registered Customer	• Consumer direct: Fashion Flow • B2B direct: ToolTech • Demand chain: Channel hub, Hosted store • Hosting: Hosting hub, Hosted store • Supply chain: Store directory, Supplier hosted store	This role is available in the samples, but does not have access to any specific tool.
Returns Administrator	• Consumer direct: Fashion Flow • B2B direct: ToolTech	• Accelerator
Sales Manager	• B2B direct: ToolTech • Supply chain: Supplier hosted store	• Accelerator
Seller	• Consumer direct: Fashion Flow • B2B direct: ToolTech • Demand chain: Hosted store • Hosting: Hosted store • Supply chain: Supplier hosted store	• Accelerator
Seller Administrator	• Consumer direct: Fashion Flow • B2B direct: ToolTech • Demand chain: Channel hub, Hosted store • Hosting: Hosting hub, Hosted store • Supply chain: Store directory, Supplier hosted store	• Organization Administration Console

Table 1. Roles mapped to WebSphere Commerce tools (continued)

Roles	Samples	Tools
Site Administrator (Root Organization)	• Consumer direct: Fashion Flow • B2B direct: ToolTech • Demand chain: Channel hub, Hosted store, Catalog asset store, Reseller storefront asset store • Hosting: Hosting hub, Hosted store, Catalog asset store, Hosted storefront asset store • Supply chain: Store directory, Supplier hosted store, Catalog asset store, Supplier asset store	• Accelerator • Organization Administration Console • Administration Console

Notes:

1. The Site Administrator is the only role with access to the Administration Console.
2. For more information on the specific roles and the menus in each tool they have access to, see the ″Roles″ file in the WebSphere Commerce Production online help.
3. For more information on each sample store, see ″ Stores″ in the WebSphere Commerce Production and Development online help

How access control prevents unauthorized actions

This section explains how policy-based access control works to ensure that users can perform only actions for which they are authorized.

Checking for authorization before performing a user-initiated action

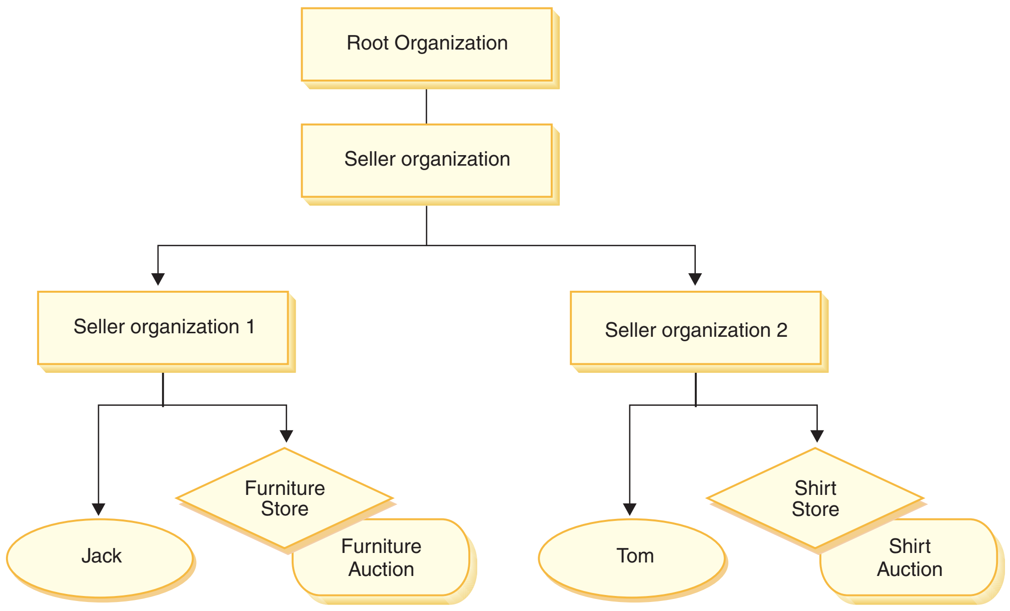
Policy Manager is the access control component that determines whether or not the current user is allowed to execute the specified action on the specified resource. Access control policies are specified in XML format. During instance creation, the default policies and policy groups are loaded into the appropriate database tables. When WebSphere Commerce Application Server is started up, the access control information is cached in memory so the Policy Manager can quickly check a user's authorization when called to do so. If access control information is changed in the database through the WebSphere Commerce Administration Console, or by loading XML policy data, the access control cache needs to be updated. This can be done by updating the appropriate registry in the WebSphere Commerce Administration Console. If policy data has changed, then the Access Control Policies registry should be updated. If policy group data has changed, then the Access Control Policy Groups registry should be updated. Restarting WebSphere Commerce will also result in updating the cache.

When a user attempts to perform an action on a protected resource, an access control check will be done to make sure that the user is authorized. The Policy

Manager looks for all the access control policies that apply to the organization that owns the resource. Then it checks those policies to evaluate if the user is authorized to perform the action on the target resource. If there is at least one such policy, the Policy Manager grants access, otherwise, access is denied.

Levels of access control

There are two broad levels of access control in WebSphere Commerce: command level (also know as role-based) and resource level (also known as instance-level).



Command-level or role-based access control

Command-level or role-based access control is coarse access control. It determines "who can do what". With role-based access control, you can specify that all users of a particular role can execute certain commands. Consider the access control policy, Sellers can execute sellers commands. In this policy, one of the sellers commands is the `ModifyAuction` command. In the figure above, Jack and Tom both are sellers, so both of them can modify auctions.

Role-based access control is used for controller commands and views. This type of access control does not consider the data resource that the command acts upon. It only determines if the user is allowed to execute a particular controller command or view. This level of access control is mandatory and is enforced by the run time.

Command-level access control for controller commands: Whenever you run a controller command, an access control policy must exist that grants users to perform the `Execute` action on the command resource. The resource is the interface name of the controller command. The access group is usually geared to a single role. For example, you can specify that users with the Account Representative role can execute any command in the `AccountRepresentativesCmdResourceGroup` resource group.

Command Level Access Control for Views: When a view is called directly from the URL, or is the result of a redirect from a command, it must have an access control policy. Such a policy must have the `viewname` specified as an action, in the `ACACTION` table. This action must then be associated with an action group, using the

`ACACTACTGP` table. This action group must then be referenced in the appropriate command level policy, in the `ACPOLICY` table.

Instance-level or resource-level access control

Instance-level or resource-level access control policies provide granular access control, determining `who can do what command on which resources`. The previous example of a role-based access control policy that allows Sellers to modify auctions, can be fine-tuned for resource-level access control to be, Sellers can modify auctions owned by the organization for which they play their role. In 33, Jack has the seller role for Seller Organization 1. Tom has the seller role for Seller Organization 2. Jack creates a furniture auction at the furniture store. Tom creates a shirt auction at the shirt store. Jack can modify the furniture auction, but *not* the shirt auction. Tom can modify the shirt auction, but *not* the furniture auction.

To summarize, first the system does a command-level access check. If the user is allowed to execute a command, a subsequent resource-level access control policy is done to determine if the user can access the resource in question.

Resource level access control applies to commands and databeans.

Resource-level access control for commands: After the command level access control checking has been completed, if access has been granted, then resource level checking is done in one of the following two cases:

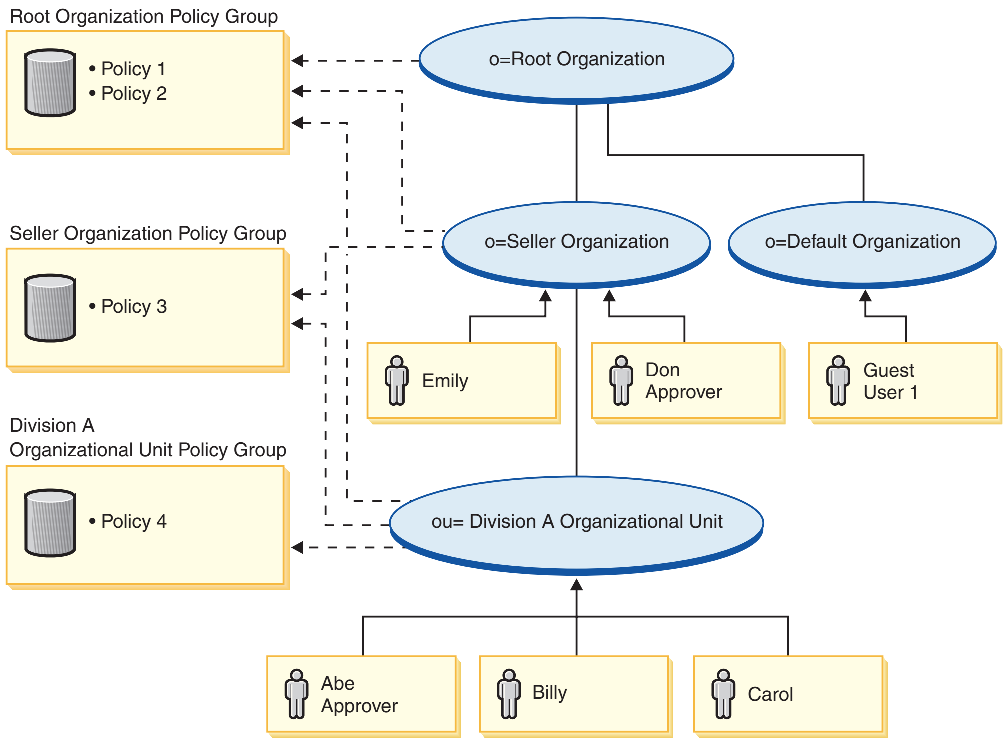
- The command implements `getResources()` — this method specifies the instances of resources that need to be checked against the current action; where the command is now the action. The WebSphere Commerce Runtime will enforce that the current user has access to all of the resources specified by `getResources()`. By default, `getResources()` returns null, that is, it does not perform any resource level checking.
- The command calls `checkIsAllowed(Object Resource, String Action)` — in cases where the command writer does not know which resources need to be checked at the time that getResources() is called by the Runtime, the command can call this `checkIsAllowed()` method, as needed, to determine if the current action and resource pair is authorized. The action is usually the interface name of the current command. When this method is called, if access is denied, an exception will be thrown: `ECApplicationException( ECMessage._ERR_USER_AUTHORITY, ..)`

Resource level access control for databeans: As explained above, views are protected by command level policies, which are usually based on roles. For example, the command level policy may specify that a Seller Administrator has access to a specific view. It is often necessary to further ensure that the databeans on the JSP are all related to the organization for which the user plays the Seller Administrator role. This is done by having all databeans that need protection (directly or indirectly), implement the Delegator interface. These databeans delegate to a primary (independent) databean which in turn implements the Protectable interface. A primary databean would delegate to itself, and therefore implement both interfaces. Then, whenever a databean is invoked using the Databean Manager's `activate()` method, the WebSphere Commerce Runtime will ensure that there is a policy which grants the current user the authority to perform the `Display` action on the primary databean resource.

Evaluating access control policies

This section can be used as a guide to evaluating access control policies. In this section, you are presented with a scenario and guided through an example of how to evaluate groupable standard and groupable template access control policies. Each section begins with a description of related policies, and scenarios using each policy. For more information on groupable standard and groupable template policies, see “Types of access control policies” on page 27.

The following diagram graphically displays the scenario:



Organizational hierarchy

From the diagram, you can see the following organizations are in the site:

- Root Organization
- Seller Organization
- Default Organization
- Division A Organization Unit

The solid lines in the diagram indicate ownership, the dotted lines indicated subscriptions. As you can see, Root organization is the parent of Seller organization, and Default organization. Seller Organization is the parent of Division A Organization Unit.

Users

In the diagram, Don and Emily are registered to the Seller Organization. Abe, Billy and Carol are registered to Division A Organization Unit. Guest user 1 has not registered, but for access control purposes, implicitly belongs to the Default Organization.

Roles

Don has the approver role for the Seller Organization. Abe has the approver role for the Division A Organization Unit.

Access Groups

The following access groups are used in this scenario:

- Registered users: This group implicitly includes all users that are registered to at least one organization in the site.
- Approvers for Seller: This group implicitly includes all users that have the role of approver for the Seller organization.
- Approvers for Division A: This group implicitly includes all users that have the role of approver for the Division A Organization Unit.

Documents

The document object is a protected resource. The owner of a document is defined to be the organization where it was created.

Access control requirements for updating documents

The following are the access control requirements for updating documents:

1. Registered users can update a document of which they are the creator.
2. Approvers for Division A can update documents owned by Division A, but not documents owned by Seller. Approvers for Seller organization can update documents owned by both Division A, and Seller organization.

Evaluating groupable standard policies

This section guides you through the groupable standard policies and the scenarios to evaluate them.

Access control polices related to updating documents

The following is the policy format and the access control policies that relate to updating documents:

Policy Format: `[Access Group, Action Group, Resource Group, Relationship]`

Policy 1:

```
[Registered Users, Execute Command Action Group,  Update Document
Resource Group, - ]
```

This is a groupable standard role-based policy that is part of the Root Organization policy group to which Root Organization, Seller Organization and Division A Organizational Unit are subscribing. In this policy, registered users can execute `Update Document` commands.

Policy 2:

```
[Registered Users, Update Document Action Group, document, creator ]
```

This is a groupable standard resource-level policy that is part of the Root Organization policy group to which Root Organization, Seller Organization and Division A Organizational Unit are subscribing. In this policy, registered users can update a document if they are the creator of that document.

Policy 3:

```
[Approvers for Seller, Update Document Action Group, document, - ]
```

This is a groupable standard resource-level policy that is part of the Seller Organization policy group to which Seller Organization and Division A Organizational Unit are subscribing. In this policy, approvers for Seller can update documents that are owned by Seller.

Policy 4:

```
[Approvers for Division A, Update Document Action Group, document, - ]
```

This is a groupable standard resource-level policy that is part of Division A Organization Unit policy group to which Division A is subscribing. In this policy, Approvers for Division A can update documents that are owned by Division A.

Scenarios

Scenario 1 : Billy attempts to update his own document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root Organization. So, only policies belonging to the policy groups subscribed to by Root Organization will be used to evaluate whether the user has command-level access: policies 1 and 2 are part of the policy group that to which Root Organization is subscribing.
2. Policy 1 grants access, since Billy is a member of the Registered Users access group and he is performing the `Execute` action on the `Update Document` command resource.

Resource - level check:

1. The `Update Document` command specifies that the document resource is to be protected. Billy's document is owned by Division A. Since Division A subscribes to policy groups, all policies belonging to those policy groups will apply: policies 1, 2, 3 and 4.
2. Policy 2 grants access since Billy is a member of the Registered Users access group, he is performing the `Update Document` command action on the document resource, and he fulfills the creator relationship with the document.

Since Billy passed both the command-level and resource-level access control checks, he can update his own document.

Scenario 2: Don attempts to update Carol's document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root organization. So, only policies belonging to the policy groups subscribed to by Root Organization will be used to evaluate whether the user has command-level access: policies 1 and 2 are owned by Root organization.
2. Policy 1 grants access, since Don is a member of the Registered Users access group and he is performing the `Execute` action on the `Update Document` command resource.

Resource - level check:

1. The `Update Document` command specifies that the document resource is to be protected. Carol's document is owned by Division A. Since Division A subscribes to policy groups, all policies belonging to those policy groups will apply: policies 1, 2, 3 and 4.
2. Policy 3 grants access since Don is a member of the `Approvers for Seller` access group, and he is performing the `Update Document` command action on the document resource.

Since Don passed both the command-level and resource-level access control checks, he can update Carol's document.

Scenario 3: Abe attempts to update Emily's document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root organization. So, only policies belonging to the policy groups subscribed to by Root Organization will be used to evaluate whether the user has command-level access: policies 1 and 2 are owned by Root organization.
2. Policy 1 grants access, since Abe is a member of the Registered Users access group and he is performing the `Execute` action on the `Update Document` command resource.

Resource - level check:

1. The `Update Document` command specifies that the document resource is to be protected. Emily's document is owned by Seller organization. Since Seller Organization subscribes to policy groups, all policies belonging to those policy groups will apply: policies 1, 2 and 3.
2. Policy 3 does NOT grant access since Abe is NOT a member of the Approvers for the Seller access group.

Although Abe passed the command-level check, since he failed the resource-level access control check, he cannot update Emily's document.

Scenario 4: Guest user 1 attempts to update his own document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root Organization. So, only policies belonging to the policy groups subscribed to by Root Organization will be used to evaluate whether the user has command-level access: policies 1 and 2 are owned by Root Organization.
2. Policy 1 does NOT grant access, since Guest user 1 is NOT a member of the `Registered Users` access group.

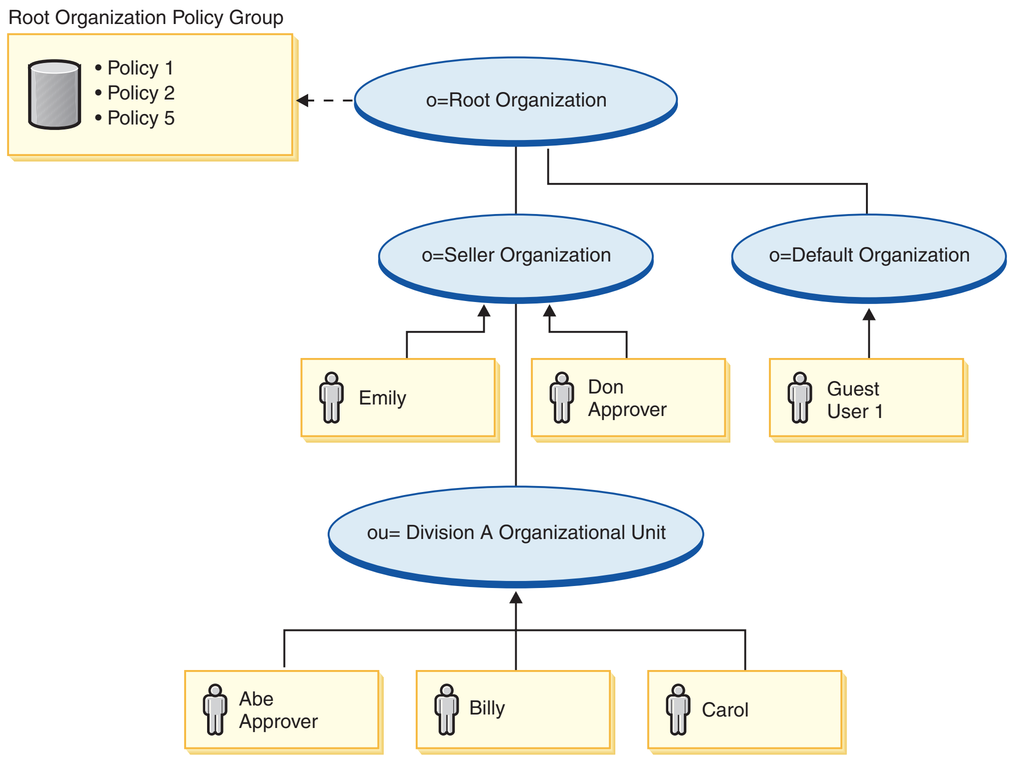
Resource - level check:

1. Resource-level checking is not done since the command-level check failed

Since Guest user 1 failed the command-level check, he cannot update his own document.

Evaluating groupable template policies

This section is based on the configuration shown in the following diagram.



Access control policies related to updating documents

In this configuration, access control policies 1 and 2 still apply, however, groupable standard policies 3 and 4 are now replaced by groupable template policy 5. For more information on policies 1 and 2 see, "Evaluating groupable standard policies" on page 36.

Policy 5:

```
[Approvers for Organization, Update Document Action Group, document, - ]
```

This policy is a groupable template resource-level policy. It is part of the Root Organization policy group to which Root Organization is subscribing. Groupable template policies dynamically apply to the organization that owns the resource during run time. These policies typically use parameterized access groups. In this case, the following parameterized access group is used:

- Approvers for Organization: This group implicitly includes all users that have the role of approver for the organization that owns the document resource or for its ancestor organizations.

Scenarios

The following scenarios are based on the configuration shown in the previous diagram which has only one policy group. Root Organization policy group includes policies 1, 2, and 5.

Scenario 1: Don attempts to update Carol's document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root organization. So, only policies belonging to policy groups subscribed by Root Organization will be used to evaluate whether the user has command-level access: policies 1, 2, and 5.
2. Policy 1 grants access, since Don is a member of the Registered Users access group and he is performing the `Execute` action on the `Update Document` command resource.

Resource - level check:

1. The `Update Document` command specifies that the document resource is to be protected. Carol's document is owned by Division A. Division A does not subscribe to any policy groups, so the access control framework will begin searching up the organization hierarchy until it encounters an organization that subscribes to at least one policy group. Division A's immediate parent organization, Seller Organization, also does not subscribe to policy groups. Continuing up the organization hierarchy, Root Organization is reached. This organization subscribes to a policy group; thus its policies can be applied: policies 1, 2, and 5.
2. Groupable template policy 5 is applied to the organization that owns the resource: Division A. The parameterized access group, Approvers for Organization, dynamically scopes to the current resource context such that it will check if the user satisfies the access group condition for the organization that owns the resource or its ancestors. In this case, Don is an approver for Seller Organization (an ancestor of Division A), so he satisfies the conditions of the access group. Since he is performing the Update Document command action on the document resource, the other elements of policy 5 are also satisfied, so the resource-level policy check passes.

Since Don passed both the command-level and resource-level access control checks, he can update Carol's document.

Scenario 2: Abe attempts to update Emily's document: The following is the access control evaluation for this scenario:

Command - level check:

1. There is no store ID specified, so the owner of the command is set to Root organization. So, only policies belonging to policy groups subscribed by Root Organization will be used to evaluate whether the user has command-level access: policies 1, 2, and 5.
2. Policy 1 grants access, since Abe is a member of the Registered Users access group and he is performing the `Execute` action on the `Update Document` command resource.

Resource - level check:

1. The`Update Document` command specifies that the document resource is to be protected. Emily's document is owned by Seller Organization. Seller Organization does not subscribe to any policy groups, so the access control framework will begin searching up the organization hierarchy until it encounters an organization that subscribes to at least one policy group. Continuing up the organization hierarchy, Root Organization is reached. This organization subscribes to a policy group; thus its policies can be applied: policies 1, 2 and 5.
2. Groupable template policy 5 is applied to the organization that owns the resource: Seller Organization. The parameterized access group, Approvers for Organization, dynamically scopes to the current resource context such that it

will check if the user satisfies the access group condition for the organization that owns the resource or its ancestors. In this case, Abe is an approver for Division A Organization Unit (a descendant of Seller Organization), so he does not satisfy the conditions of the access group.

Although Abe passed the command-level check, since he failed the resource-level access control check, he cannot update Emily's document.

Looking at a policy in detail

Now that we understand the basic structure of an access control policy and the types of policies there are, let us look at one of the default policies in detail, using a series of different examples. The policy we will study is the following:

`AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`

Note: This policy is a resource-level policy. Its' policy type is groupable template.

In the first example, we will learn how to read the policy using the WebSphere Commerce Organization Administration Console, identify its parts, and understand what the policy means. The second example will look at the policy in XML, to help you understand what the same information looks like in the code.

The third example goes a step further in understanding how one policy is related to other policies. Understanding dependencies between policies is an important prerequisite for making changes to access control policies, or creating new ones.

Example 1: Reading a policy

In this example, we will use the WebSphere Commerce Organization Administration Console to look up a policy and identify the parts that define it. We will also use these parts to form a general description of the policy.

Looking up the policy in the Organization Administration Console

1. Log in to the WebSphere Commerce Organization Administration Console. From the Access Management menu, select **Policies**.
2. Select Root Organization from the list box, since Root Organization owns most of the default access control policies.
3. On the Policies page, scroll through the list of policies and locate the following policy:
   `AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`
   Notice that you can scroll through the list of policies by using the scroll bar as well as using the **First**, **Previous**, **Next** and **Last** links.

Viewing the parts of the policy

1. Select the policy by clicking the box next to it and click **Show Action Group**.
2. On the Action Group page, you will see the action group, AuctionManage. This is the action group associated with the policy. Select `AuctionManage` and click **Show Actions**.
3. On the next page, you will see the following list of actions, or commands, included in the `AuctionManage` action group:
   - `com.ibm.commerce.negotiation.commands.CloseBiddingCmd`
   - `com.ibm.commerce.negotiation.commands.DeleteAuctionCmd`

- `com.ibm.commerce.negotiation.commands.ModifyAuctionCmd`

  Here, `AuctionManage` includes closing an auction (`CloseBiddingCmd`), deleting an auction, (`DeleteAuctionCmd`), and modifying an auction (`ModifyAuctionCmd`). For more information on commands, refer to the reference section in the online help documentation.

  Notice that you can also access the same list of actions from the Policies page by clicking **Show Actions**.

4. To return to the policies page, select any of the actions, and click **Show Policies**.
5. Select the policy again, but now click **Show Member Group** to see the member (access) group that is used in this policy.
6. Make a note of the member (access) group name. In this case, the member (access) group is `AuctionAdministratorsForOrg`.
7. From the Access Management menu, select **Access Groups**.
8. Find `AuctionAdministratorsForOrg`. Select it and click **Change.**
9. Click **Criteria**. On the Criteria page, look under Selected roles and organizations. You should see the following roles:
   - `Seller-For organization`
   - `Product Manager-For organization`
   - `Buyer (sell-side)-For organization`
   - `Category Manager-For organization`

   Any user assigned one of these roles for the organization that owns the auction resource, is part of the `AuctionAdministratorsForOrg` access group.
10. Leave the Criteria page without making any changes. From the Access Management menu, select **Policies** again. Locate the following policy:

    `AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`
11. Select the policy and Click **Show Resources**. On the Resources page, you will see the `com.ibm.commerce.negotiation.objects.Auction` resource. This is the resource on which the actions listed in the action group act. In this case, the resource is an auction. Notice that you can access this same list from the Policies page by clicking **Show Resource Group** and drilling down to the individual resources.
12. Now select **Policies** from the Access Management menu, and locate the following policy:

    `AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`
13. Select the policy and click **Change**. On the Change Policy page, look at the drop-down menu under **Relationship**. Note that the relationship is set to none. This means that the policy does not have a relationship.
14. Click **Cancel** and **OK** to the dialog box.

Understanding what the policy means

Now that we have identified the individual parts of this policy, we can start to piece them together to understand what the policy does. First, we know that the policy applies to all users belonging to the `AuctionAdministratorsForOrg` group. We learned this by clicking **Show Member Group**. From there we used the Access Management menu to go to the Access Group page, and saw that the access group included the following roles: `seller`, `product manager`, `buyer (for the sell-side)`, and the `category manager`. Collectively, users with one of these four roles can be referred to as an Auction Administrator.

We also know that the action group contains the commands for modifying, retracting, and closing an auction, and that the resource group includes only the auction resource that is being managed. Again, we know this by clicking **Show Actions** and **Show Resources** from the Policies page and drilling to the detail level. Lastly, we can tell that the policy does not include a relationship between the access group and the resources.

Putting everything together, we can conclude that this policy permits Auction Administrators to perform all the activities associated with managing auctions on an auction resource, such as modifying, retracting, and closing an auction, as long as the administrator plays the role for the organization that owns the auction.

We can get a sense of what a policy means by looking at its name. In this example, the policy starts with the name of the designated group of users, `AuctionAdministratorForOrg`. The notation, `ForOrg`, indicates that this is a groupable template policy. `AuctionManageCommands` describes the action group, and `AuctionResource` describes the resource group.

Example 2: Reading a policy in XML

The default access control policies are stored in an XML file that is loaded into your database during instance creation. When you look at a policy in the WebSphere Commerce Administration Console, you are using the interface to view and make changes to the information stored in the database. The information in the database is used by the Policy Manager to evaluate access control. If the database information is more recent than the XML file, you can use the Extractor tool to extract the access control policy information from the database into an XML file.

This is what a policy looks like in the XML file:

```
<!-- AuctionAdministrators
manage Auctions (Retract/delete auction,
Modify auction, Close Auction)
-->
<Policy
Name="AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource"
OwnerID="RootOrganization"
UserGroup="AuctionAdministratorsForOrg"
ActionGroupName="AuctionManage"
ResourceGroupName="AuctionDataResourceGroup"
PolicyType="groupable Template">
</Policy>
```

Here, the policy is defined by the following:

`Name`: The name of the policy.

`OwnerID`: The organization for which the policy applies.

`UserGroup`: The access group.

`ActionGroupName`: The action group.

`ResourceGroupName`: The resource group.

`PolicyType`: The type of policy, such as groupable standard or groupable template.

The file that contains all of the default access control policies is called `defaultAccessControlPolicies.xml` and is located in the following directory:

`X:\installation_directory\xml\policies\xml`.

Note: The descriptions for each default access control file are contained in the `defaultAccessControlPolicies_locale.xml` file, which can be found in the same directory. A change made to a default access control policy in the default access control file, needs to have its corresponding description updated in `defaultAccessControlPolicies_en_US.xml`. It is strongly recommended that any changes made to the XML files be reserved for advanced users.

Example 3: Identifying other policies associated with your policy

In this last example, we will take a look at how an access control policy can have dependencies on other policies.

Policies that define the commands (actions) that a group of users (an access group) can perform on a resource are called resource-level policies. For example, the policy we have been looking at in detail:

`AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource` is an example of a resource-level policy.

However, the actions permitted by a resource-level policy are also dependent on the actions permitted for each role belonging to the policy's access group. Policies that describe what actions are permitted for a particular role are called role-based policies.

To identify the role-based policies associated with a resource-level policy, do the following:

Looking up the roles associated with the policy

1. Log in to the WebSphere Commerce Administration Console and locate the resource-level policy on the Policies page. Using the same example, we know the policy we want is the following:
   `AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`

   .
2. Identify the access group associated with the policy. In this case, we already know that the access group is `AuctionAdministratorsForOrg`.
3. Look up the roles associated with the access group. For `AuctionAdministratorsForOrg`, we know from the previous examples that the roles are: `Buyers (sell-side)`, `Category Managers`, `Product Manager`, and `Sellers`.

Looking up the role-based policies for each role

1. Turn to the Appendix at the end of this book and find the section heading, Role-Based Policies. You will use the Appendix to locate each role-based policy associated with a role.
2. Find the `Buyers(sell-side)ExecuteBuyers(sell-side)CommandsResourceGroup` policy. This policy is associated with the `Buyers (sell-side)` role. We know this because of the `Buyers(sell-side)` prefix to the policy.

3. Find the rest of the role-based policies associated with `Buyers (sell-side)`, `Category Manager`, `Product Manager`, and `Sellers` roles, using their prefixes to identify the right policies. You should come up with the following list:
   - `Buyers(sell-side)ExecuteBuyers(sell-side)CommandsResourceGroup`
   - `Buyers(sell-side)ExecuteBuyers(sell-side)Views`
   - `CategoryManagersExecuteCategoryManagersCmdResourceGroup`
   - `CategoryManagersExecuteCategoryManagersViews`
   - `ProductManagersExecuteProductManagersCmdResourceGroup`
   - `ProductManagersExecuteProductManagersViews`
   - `SellersExecuteSellersCmdResourceGroup`
   - `SellersExecuteSellersViews`
4. Each role-based policy permits users with that role to carry out particular controller commands or views. To see what actions and resources are associated with a role-based policy, look up the policy on the Policies page from the WebSphere Commerce Organization Administration Console, using the same procedure from Example 1.

Why identifying dependencies between policies is important

Understanding which role-based policies are associated with a resource-level policy is often a prerequisite for customizing your policies, and for creating new ones.

In Part 3, “Administering security authorization,” on page 83, you will learn more about resource-level and role-based policies, including how to recognize them, understand their differences, and see how they are related to each other.

Part 2. Administering security authentication

This part describes the security authentication tasks that can typically be performed by the WebSphere Commerce site administrator.

Chapter 4. Enhancing site security

To enhance the security of your WebSphere Commerce site, you can enable any of the following features in WebSphere Commerce Configuration Manager:

- Log off a user that is inactive for an extended period and request they log back on to the system, using the Login Timeout node. For details, see "Enabling login timeout" on page 53.
- Require users to change their passwords when they are logging in to the system for the first time, using the Password Invalidation node. For details, see "Activating password invalidation" on page 53.
- Require users to enter their passwords if they are running requests that run designated commands, using the Password Protected Commands node. For details, see "Enabling password protected commands" on page 54.
- Update encrypted data such as passwords and credit card information as well as the merchant key in a WebSphere Commerce database, using the Database Update Tool node. For details, see "Updating encrypted data" on page 55.
- Reject any user request that contain attributes or characters that are designated as not allowed, using the Cross Site Scripting Protection node. For details, see "Enabling cross site scripting protection" on page 55.
- Quickly identify any security threats against WebSphere Commerce by enabling access logging. For details, see "Enabling access logging" on page 58.

In addition, you can enable the following features from the Security drop-down in the WebSphere Commerce Administration Console:

- Set up an account policy for your site to define the account-related policies in use, by using the Account policy page. For details, see "Setting up an account policy" on page 59.
- Set up a password policy for your site to control a user's password selection characteristics using the Password policy page (only if users are authenticated against the WebSphere Commerce database). For details, see "Setting up a password policy" on page 59.
- Set up an account lockout policy for your site to reduce the chances of a user account being compromised, using the Account lockout policy page (only if users are authenticated against the WebSphere Commerce database). For details, see "Setting up an account lockout policy" on page 61.
- Launch a security program that checks and deletes temporary WebSphere Commerce files that may contain potential security exposures using the Launch security check page. For details, see "Launching a security check" on page 61.

For information on related concepts, see the following topics in the WebSphere Commerce online help:

- Configuration Manager
- WebSphere Commerce configuration file
- Administration Console
- Security

For information on related tasks, see the following topics in the WebSphere Commerce online help.

- Launch the Configuration Manager

- Open the Administration Console

Security consideration for the Internet Information Services (IIS) Web server

Attention

If you are using the IIS Web server with WebSphere Commerce, you need to be aware of the following security consideration and take the recommended action to minimize any security exposure of your WebSphere Commerce data.

Problem: For the IIS Web server, read permission on a Virtual Directory provides access to the source code of JSP files. In order to prevent download of the JSP source code, you should must physically separate the static content from the dynamic content of your Web pages, if you are using the IIS Web server. This is because IIS security is based on directory location, rather than file type. Under the default IIS configuration, the image files and JSP files are located under a single alias. You should use the default configuration for testing purposes only.

Solution: To secure all Web assets, *the dynamic content must be accessed using a Virtual Directory with execute only permissions (not read) while static content should be moved to a different Virtual Directory with read only permission*. For further information on how to set permissions on a Virtual Directory, see the instructions in the IIS help information. It is also recommended that you consult Microsoft® Corporations's current documentation on security patches and configuration policies.

Views for security

Before using certain security features of WebSphere Commerce, you are required to define the associated views for your store before you can use that feature. The following information describes how to define the views for:

- Login timeout (see "Login timeout")
- Password invalidation (see "Password invalidation" on page 51)
- Password protected commands (see "Password protected commands" on page 52)
- Cross site scripting protection (see "Cross site scripting protection" on page 52)

For general information on creating views and developing your store front, see the *WebSphere Commerce Store Development Guide*.

Login timeout

To use the login timeout security feature, you need to define the LoginTimeoutErrorView and ReLogonFormView views for your store.

LoginTimeoutErrorView

If the login timeout information is incorrect,WebSphere Commerce redirects the user's browser to this view. If this occurs, it is likely because someone has tampered with the cookie.

Table 2. LoginTimeoutErrorView attributes

ECConstants.EC_LOGIN_TIMEOUT_ERROR_MSGCODE	**1**	Expiry time is set to wrong value.
	2	Logon time is set to wrong value.
	3	Expiry or logon time set to wrong value.

ReLogonFormView

This view is displayed to users after their session has expired. It needs to provide the user with a form to enter the user's logon ID and password. The submit button will invoke the Logon command. There should also be a Cancel button to redirect the user to another page, in most cases, the store front page.

There are no attributes for ReLogonFormView.

Table 3. ReLogonFormView form attributes

ECUserConstants.EC_UREG_LOGONID	The user's logon id.
ECUserConstants.EC_UREG_LOGONPASSWORD	The user's logon password.
ECUserConstants.EC_RELOGIN_URL	The URL that is displayed if the credentials provided are invalid. In most cases, it will be name of this view.
ECConstants.EC_STORE_ID	The store identifier.
ECConstants.EC_URL	The URL that is displayed when the credentials that are entered belong to different user. In most cases, this should be a store home page, or the same URL that is used in a store logon page.

Password invalidation

To use the password invalidation security feature, you need to define the ChangePassword view for your store.

ChangePassword

This view is displayed if a user's password has expired. It should provide the user with a form to enter the current (expired) password and a new password. The Submit button invokes the ResetPassword command. There should also be a Cancel button that redirects the user to another page, in most cases, the store front page.

Table 4. ChangePassword attributes

ECConstants.EC_PASSWORD_EXPIRED_FLAG	**1**	The user's password has expired. This attribute is required in order to distinguish this view from the view used for the password change feature as they are the same. The view for the password change could be invoked by a user, and the JSP assigned to this view should be the same for both cases. The JSP should look for this attribute in order to decide what to display.
	null	The attribute is not on a URL . This is normal password change behavior
ECUserConstants.EC_UREG_LOGONID	The current user logon id.	
ECConstants.EC_LOGIN_RETURN_URL	The URL to which the browser is redirected after a successful password change. This URL will be passed to an action command under the name ECConstants.EC_URL.	

Table 5. ChangePassword form attributes

ECUserConstants.EC_UREG_LOGONID	The logon ID id of the user. The current logon ID has been passed in to the view.
ECUserConstants.EC_UREG_LOGONPASSWORDOLD	The old password.
ECUserConstants.EC_UREG_LOGONPASSWORD	The new password.
ECUserConstants.EC_UREG_LOGONPASSWORDVERIFY	The new password verification.

Table 5. ChangePassword form attributes (continued)

ECConstants.EC_URL	The URL where users are redirected after a successful password change. The value has been passed in to the view.
ECUserConstants.EC_RELOGIN_URL	The URL where the browser is redirected if the password change is not successful.

Password protected commands

To use the password protected commands security feature, you need to define the PasswordReEnterErrorView and the PasswordReEnterFormView views for your store.

PasswordReEnterErrorView

This view is used in the following scenarios:

- A user fails to provide the correct password and is logged off.
- The authentication has failed.

In both cases, the user should have a way to continue to another page through a link on the current page.

Table 6. PasswordReEnterErrorView attributes

ECConstants.EC_PASSWORD_REREQUEST_MSGCODE	**0**	A problem occurred when attempting to authenticate the user.
	null	The attribute is not on a URL. The user failed to provide the password is logged off.

PasswordReEnterFormView

This view is displayed when user tries to execute a password protected command. It should provide user with form to enter password. There should be two entry fields for password.

Table 7. PasswordReEnterFormView attributes

ECConstants.EC_PASSWORD_REREQUEST_URL	The URL is run using the Submit button of the form.	
ECConstants.EC_PASSWORD_REREQUEST_MSGCODE	The message code specifying the message that is shown to the user:	
	1	The passwords that were entered do not match.
	2	A password was not entered.
	3	An incorrect password was entered.

ACTION: The URL is passed in as a parameter named:

Table 8. PasswordReEnterFormView form attributes

ECConstants.EC_PASSWORD_REREQUEST_PASSWORD1	The first password.
ECConstants.EC_PASSWORD_REREQUEST_PASSWORD2	The second password.

Cross site scripting protection

To use the cross site scripting security feature, you need to define the ProhibitedAttrsErrorView, ProhibitedCharacterErrorView, and ProhibCharEncodingErrorView views for your store.

ProhibitedAttrsErrorView
This view is shown to the user when the request is not processed because it contained prohibited attributes.

ProhibitedCharacterErrorView
This view is shown to the user when the request is not processed because it contained prohibited characters

ProhibCharEncodingErrorView
This is the same as ProhibitedCharacterErrorView, above.

Enabling login timeout

Note: To use the login timeout security feature for a store, you need to define the LoginTimeoutErrorView and ReLogonFormView views for the store as described in "Login timeout" on page 50.

Use the Login Timeout node of the Configuration Manager to enable or disable the login timeout feature. When this feature is enabled, a WebSphere Commerce user that is inactive for an extended period of time is logged off the system and requested to log back on. If the user subsequently logs on successfully, WebSphere Commerce runs the original request that was made by the user. If the user logon fails, the original request is discarded and the user remains logged off the system.

Note that for WebSphere Commerce tools (such the Administration Console,WebSphere Commerce Accelerator, and so on), login timeout does not present a relogin page to the user. Instead, it closes the browser window and it is up to the user to log back on to the tool. Thus, in the case of tools, the original request that the user submits is not processed.

To enable this feature:

1. Launch the Configuration Manager and traverse to the Login Timeout node for your instance as follows: **WebSphere Commerce >** *host_name* **> Instance List >** *instance_name* **> Instance Properties> Login Timeout**
2. To activate the login timeout feature, click the **Enable** check box.
3. Enter the login timeout value, in seconds, in the Value field.
4. To apply your changes to Configuration Manager, click **Apply**.
5. Upon successfully updating the configuration for your instance, you will receive a message indicating a successful update.
6. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Note that the login timeout value is stored in the *instance*.xml file in milliseconds, while the value in Configuration Manager is entered in seconds.

Activating password invalidation

Note: To use the password invalidation security feature, you need to define the ChangePassword view for your store as described in "Password invalidation" on page 51.

Use the Password Invalidation node of the Configuration Manager to enable or disable the password invalidation feature. Password invalidation, when enabled, requires WebSphere Commerce users to change their password if the user's password has expired. In this case, the user is redirected to a page where they are required to change their password. Users are not able to access any secure pages on the site until they have changed their password. To enable this feature:

1. Launch the Configuration Manager and traverse to the Password Invalidation node for your instance as follows: **WebSphere Commerce >** *host_name* **> Instance List >** *instance_name* **> Instance Properties > Password Invalidation**
2. To activate the password invalidation feature, click the **Enable** check box.

3. To apply your changes to Configuration Manager, click **Apply**.
4. Upon successfully updating the configuration for your instance, you will receive a message indicating a successful update.
5. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Enabling password protected commands

Note: To use the password protected commands security feature, you need to define the PasswordReEnterErrorView and the PasswordReEnterFormView views for your store as described in "Password protected commands" on page 52.

Use the Password Protected Commands node of the Configuration Manager to enable or disable the password protected commands feature. When this feature is enabled, WebSphere Commerce requires registered who are logged onto WebSphere Commerce to enter their password before continuing a request that runs designated WebSphere Commerce commands.

Caution: When you configure password protected commands, some of the commands shown in the command selection list can be executed by generic or guest users. Configuring such commands as password protected will restrict generic and guest users from running them. Therefore, you should exercise caution when you configure commands to be password protected.

To enable this feature:

1. Launch the Configuration Manager and traverse to the Password Protected Commands node for your instance as follows: **WebSphere Commerce>** *host_name* **> Instance List >** *instance_name* **> Instance Properties > Password Protected Commands**
2. In the General tab:
   a. To activate the password protected commands feature, click **Enable**.
   b. Enter number of retries in the Retries field. (The default number of retries is 3.)
3. In the Advanced tab:
   a. Select a WebSphere Commerce command you wish to protect from the list in the Password Protected Command List window and click **Add**. The command you have selected is listed in the Current Password Protected List window.
   b. If you wish to disable password protection for any WebSphere Commerce command, select the command in the Current Password Protected Command list window and click **Remove**.
4. To apply your changes to Configuration Manager, click **Apply**.
5. Upon successfully updating the configuration for your instance, you will receive a message indicating a successful update.
6. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Note: WebSphere Commerce will only display the commands that are designated as `authenticated` or set with the `https` flag in the `URLREG` table in the list of available commands.

Updating encrypted data

Use the Database Update Tool available from the Database node of the Configuration Manager to change the merchant key and update all encrypted data (for example, passwords or credit card numbers) in one or more WebSphere Commerce databases for a given instance. To use the tool:

1. Launch the Configuration Manager and traverse to your specific database entry as follows: **WebSphere Commerce>** *host_name* **> Instance List >** *instance_name* **> Instance Properties > Database>** *database_name*
2. Right-click on *database_name* and select **Run Database Update Tool**
   - Select **Update all databases for this instance** to migrate encrypted data for all databases for the selected instance.

     400 As iSeries supports a single database configuration, this option does not apply to iSeries.
   - Select **Update selected database** to migrate encrypted data for a specific database by selecting the database from the drop-down list (default).
3. Select an action you want to run from the Action Item box, and fill in the required information in the Parameter field:

Actions	**Parameters**	**Action Required**
Change Merchant Key	Old Merchant Key	Enter your existing merchant key that you used when you created your current WebSphere Commerce instance.
	New Merchant Key	Enter your new merchant key. This is a 16-digit hexadecimal number for the Configuration Manager to re-encrypt the currently encrypted data. The Merchant Key must have at least one alphanumeric character (a to f) and at least one numeric character (0 to 9). Any alphanumeric character must be entered in lower case letters, and you cannot have the same character entered more than four times in a row.

4. Click **OK** to run the database update tool for your selected WebSphere Commerce database or for all your WebSphere Commerce databases.
5. Upon successfully updating the configuration for your instance, you will receive a message indicating a successful update.
6. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Enabling cross site scripting protection

Note: To use the cross site scripting security feature for a store, you need to define the `ProhibitedAttrsErrorView`, `ProhibitedCharacterErrorView`, and `ProhibCharEncodingErrorView` views for the store as described in "Cross site scripting protection" on page 52.

Use the Cross Site Scripting Protection node of the Configuration Manager to enable or disable cross site scripting protection for your instance. When enabled, cross site scripting protection rejects any user requests that contain attributes or strings that are designated as not allowable. You can specify the disallowed attributes and strings in this node of the Configuration Manager. You can also exclude commands from cross site scripting protection by allowing the values of specified attributes for that particular command to contain prohibited strings. Cross site scripting protection is disabled by default.

Warning: Cross site scripting protection is a restrictive feature in that it will restrict the execution of the commands based on the configuration. The feature does not check what attributes or strings have been defined as prohibited, so when you configure it, make sure that prohibited attributes are not those used by the commands. Also make sure the prohibited strings are not values that are usually passed to the commands. Use extreme caution when configuring this feature.

To enable this feature:

1. Launch the Configuration Manager and traverse to the Cross Site Scripting Protection node for your instance as follows: **WebSphere Commerce>** *host_name* **> Instance List>** *instance_name* **> Instance Properties > Cross Site Scripting Protection**
2. Use the General tab to activate the cross site scripting protection feature, as follows:
   a. Click **Enable**.
   b. To add attributes that you wish to disallow for WebSphere Commerce commands, right-click on the Prohibited Attributes table and select **Add row**. Type the attribute that you wish to disallow. You can only specify one attribute per row.
   c. To remove attributes from the Prohibited Attributes table, highlight and right-click the line containing the attribute in the table and select **Delete row**.
   d. To add strings that you wish to disallow for WebSphere Commerce commands, right-click on the Prohibited Characters table and select **Add row**. Add the string that you wish to disallow. You can only specify one string per row.
   e. To remove characters from the Prohibited Characters table, highlight and right-click the line containing the character in the Prohibited Characters table and select **Delete row**.

   Note: The following strings are specified by default in the prohibited characters fields. These strings are most commonly used as scripting tags in malicious cross site scripting attacks:
   - `<SCRIPT`
   - `&lt;SCRIPT`
   - `<%` and `&lt;%`

   .
3. Use the Advanced tab to exclude WebSphere Commerce commands from cross site scripting protection by allowing the values of specified attributes for that particular command to contain prohibited strings as follows:
   a. Select the commands from the Command List box.
   b. Type in a list of attributes, separated by commas, for which prohibited characters are allowed in the List of Excepted Attributes window and click **Add**.
   c. To remove a command along with its attributes, select the command from the List of Excepted Commands window and click **Remove**.

You can also remove specific attributes from a command by selecting the attribute and clicking **Remove**.

4. To apply your changes to Configuration Manager, click **Apply**.
5. Upon successfully updating the configuration for your instance, you will receive a message indicating a successful update.
6. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Notes:

1. When commands are excluded from cross site scripting protection, the values of specified attributes will be encoded using HTML encoding of symbols. For example, the command `cmd1?user=<Thomas>` is encoded as `ascmd1?user=&#60;Thomas&#62;`
2. When you specify the string in the prohibited characters fields, be aware that:
   - A certain sequence of characters can cause the string to be converted to a single character in compliance with URL encoding standards. For example, the string `<%bb` would be converted into a string `<`*X* where *X* is a single character which has a hexadecimal representation value of HEX '`bb`' (decimal 187). In this case the string `<%bb` will not be caught by cross site scripting protection if passed in a URL.
   - A certain sequence of characters can cause the string conversion to fail if they do not comply with URL encoding standards. For example, the string `<%gg` would cause conversion to fail since HEX '`gg`' is not a valid hexadecimal value representation. In this case, the string `<%gg` will cause an exception, resulting in no response to the URL request containing such a string, whether or not cross site scripting protection is enabled.

Example: Consider the following examples:

- Prohibited strings: `<SCRIPT`, `<%`

  Prohibited attributes: `mycomment`, `description`

Command	Status
cmd1?description=Available...	rejected
cmd2?userid=Thomas...	accepted
cmd3?mycomment=<SCRIPT>...	rejected
cmd4?password=<%...%>...	rejected

- If you wish to allow the attribute `text` of the `cmd1` command to contain prohibited strings (`<SCRIPT`, `<%`), and not for other attributes. For example, for the attribute `txt`, you can exclude `cmd1` and specify `text` as the excepted attribute.

Command	Status
cmd1?text=<SCRIPT>...	accepted
cmd1?text=<%...%>...	accepted
cmd1?txt=<SCRIPT>...	rejected
cmd1?txt=<%..%>...	rejected

Enabling access logging

When enabled, the access logging feature logs either all incoming requests to the WebSphere Commerce server or only the requests resulting in access violations. Examples of access violations are authentication failure, insufficient authority to execute a command. When enabled, access logging allows a WebSphere Commerce administrator to quickly identify security threats to the WebSphere Commerce system.

When an authentication failure or authorization failure event occurs, the following information is logged to the access log file database tables, `ACCLOGMAIN` and `ACCLOGSUB`:

- Host name of the client
- ID of the thread running the command
- User ID of the client
- Time the event occurred
- Command that was run
- Store for which the command was run
- Resource on which the operation was performed
- Result of the access control check

To enable access logging, do the following:

1. Launch the Configuration Manager.
2. Select **Host name** > **Instance** > **Instance_List** and then open the **Components** folder.
3. Select **AccessLoggingEventListener**.
4. In the General panel, activate the **Enable Component** check-box.
5. Select the Advanced panel and enable **Start**.
6. Click **Apply**.
7. Exit Configuration Manager.
8. Restart the WebSphere Application Server.

To change the size of the log file, or to specify whether all requests are logged or not, you need to manually edit the *instance*`.xml` file for your WebSphere Commerce instance located in the WebSphere Commerce instances subdirectory:

1. Open the *instance*`.xml` file for your instance in an editor.
2. Locate the following node, which is located in the `<LogSystem>/<activitylog>` node:

   ```
   <accessLogging cacheSize="aa" logAllRequests="bbbbb" />
   ```

   where:

   - *aa* is an integer value specifying the maximum number of entries that will be logged to memory before entries are written to the database. Generally a higher number will result in improved performance with respect to access logging. The default value is 32.
   - *bbbbb* is either `true` or `false`. A value of `true` means that all incoming requests are logged. A value of `false` means that only access violations are logged. To prevent excessive or unnecessary logging, a value of `false` is recommended. Use `true` only when you suspect authentication problems or security contravention at your site. The default value is `false`.

3. When you have completed your updates, save the `instance.xml` file for your WebSphere Commerce instance.
4. Restart WebSphere Application Server.

In the following example, the access logging keeps 3 entries in memory before logging entries to the database tables. In addition, it logs all incoming requests to the WebSphere Commerce server:

```
<accessLogging cacheSize="3" logAllRequests="true" />
```

Setting up an account policy

The Account Policy page of the WebSphere Commerce Administration Console allows you to set up an account policy. This page lists all existing account policies including any predefined ones supplied with WebSphere Commerce by default. An account policy defines the account-related policies such as password and account lockout policies. On this page:

- You can create a new account policy by clicking **New.**
- You can change the characteristics an existing account policy by selecting the policy in the list and clicking **Change**.
- You can delete an existing account policy by selecting the policy in the list and clicking **Delete**.

To create a new account policy:

1. Open the WebSphere Commerce Administration Console.
2. From the Security drop-down menu of the Administration Console, click **Account Policy**.
3. On the Account Policy page, click **New** to create a new account policy.
4. Enter a name for the account policy in the Name field (for example, `my_account_policy`).
5. From the Password policy menu, select a preexisting password policy.
6. From the Account lockout policy menu, select a preexisting account lockout policy.
7. Click **OK**.

Once you have created an account policy, you can assign the policy to a user. Note that you cannot delete an account policy if it is in use (that is, a user is assigned to the account policy).

Also see “Default Authentication Policies” on page 63 for additional information.

Setting up a password policy

The Password Policy page of the WebSphere Commerce Administration Console allows you to control a user’s password selection in order to define the characteristics of the password to ensure that it complies with the security policy for your site. This page lists all existing password policies including any predefined ones supplied with WebSphere Commerce by default.

A password policy defines attributes with which the password must comply. The password policy enforces the following conditions:

- Whether the user ID and password can match.
- Maximum occurrence of consecutive characters.

- Maximum instances of any character.
- Maximum lifetime of the passwords.
- Minimum number of alphabetic characters.
- Minimum number of numeric characters.
- Minimum length of password.
- Whether the user's previous password can be reused.
- You can create a new password policy by clicking **New.**
- You can change the characteristics an existing password policy by selecting the policy in the list and clicking **Change**.
- You can delete an existing policy by selecting the password policy in the list and clicking **Delete**.

To create a new password policy:

1. Open the WebSphere Commerce Administration Console.
2. From the Security drop-down menu of the Administration Console, click **Password Policy**.
3. On the Password Policy page, click **New** to create a new password policy.
4. Enter a name for the password policy in the Name field (for example, `my_password_policy`)
5. Update the following as required to modify any of the values from the default value for shoppers:
   - **Can the userID and password match?** Defines whether the userID and password can be identical or not. Select either `Yes` or `No` from the list.
   - **Maximum consecutive character types**. Defines the maximum occurrence of consecutive characters in a password. The minimum value is 2 consecutive characters. For example, with a value of 2, a user cannot enter a password such as `aaabc`.
   - **Maximum instances of any character.** Defines the maximum number of times the same character can appear in a password. The minimum value is 1 instance of a character. For example, with a value of 2, a user cannot enter a password such as `abcaabc`.
   - **Maximum lifetime of the passwords.** Defines the maximum amount of time, in days, that a password can exist. The minimum value is 1 day. After this time period, a user is prompted to change their password.
   - **Minimum number of alphabetic characters**. Defines the minimum number of alphabetic characters that need to be in a password. The minimum value is 0 alphabetic characters.
   - **Minimum number of numeric characters**. Defines the minimum number of numeric characters that need to be in a password. The minimum value is 0 numeric characters.
   - **Minimum length of password**. Defines the smallest length of a password, in characters. The minimum value is 1 character.
   - **Can the password be reused?** Defines whether a user's previous password can be reused. Select either `yes` or `no` from the list.
6. Click **OK**.

Notes:

1. You cannot delete a password policy if it is in use (that is, a user is assigned to the password policy).

2. Password policies are enforced only if users are authenticated against the WebSphere Commerce database.

Also see “Default Authentication Policies” on page 63 for additional information.

Setting up an account lockout policy

The Account Lockout Policy page of the WebSphere Commerce Administration Console allows you to set up an account lockout policy for different user roles within WebSphere Commerce. This page lists all existing account lockout policies including any predefined ones supplied with WebSphere Commerce by default. An account lockout policy disables a user account if malicious actions are launched against that account in order to reduce the chances that the actions compromise the account.

An account lockout policy enforces the following items:

- The account lockout threshold. This is the number of invalid logon attempts before the account is disabled.
- Consecutive unsuccessful login delay. This is the time period for which the user is not allowed to login, after two failed attempts to login. The delay gets incremented by the configured time delay value (for example, 10 seconds) with every consecutive login failure.

To set the account lockout policy:

1. Open the WebSphere Commerce Administration Console.
2. From the Security drop-down menu of the Administration Console, click **Account Lockout Policy**.
3. The Account Lockout Policy page lists all existing account lockout policies. On this page:
   - You can create a new policy by clicking **New.**
   - You can change the characteristics an existing policy by selecting the policy in the list and clicking **Change**.
   - You can delete an existing policy by selecting the policy in the list and clicking **Delete**.

For a new account lockout policy, in the Account Lockout Policy page:

1. Enter a name for the account lockout policy in the Name field (for example, `my_policy`).
2. Enter an account lockout threshold in the Account lockout threshold field. For example, enter 6 (for six attempts)
3. Enter the consecutive unsuccessful login delay in seconds in the Wait time field. For enter 10 (for ten seconds).
4. Click **OK**.

Notes:

1. You cannot delete an account lockout policy if it is in use (that is, a user is assigned to the account lockout policy).
2. Account lockout policies are enforced only if users are authenticated against the WebSphere Commerce database.

Launching a security check

400 This feature is not applicable on WebSphere Commerce for iSeries.

The Launch Security Check page of the WebSphere Commerce Administration Console allows you to manually launch a security program that checks and deletes temporary WebSphere Commerce files that may contain potential security exposures. Normally the security check program runs as a scheduled job and by default is set to run once a month.

To invoke the security check program:
1. Open the WebSphere Commerce Administration Console.
2. From the Security drop-down menu of the Administration Console, click **Security Checker**.
3. On the Launch Security Check page, click **Launch**.

The results of the security check, including all actions taken by the program are written to the Security check log window and to the `sec_check.log` file in the `logs` subdirectory:

AIX Linux Solaris `WC_installdir/instances/`*`instance_name`*`/logs`

Windows `WC_installdir\instances\`*`instance_name`*`\logs`

Windows On non-Windows platforms, file permissions are automatically set by WebSphere Commerce in order that sensitive files cannot be accessed by unauthorized users. On Windows platforms, you need to set the permissions manually as follows. This procedure ensures that only the Administrators group has the `read/write/execute` right in for sensitive files:
1. In Windows Explorer, right-click on the *`drive`*`:\WebSphere` folder.
2. Click **Properties** and **Security**. By default the "`Everyone`" group has the **all** permission for this folder.
3. Click **Add**.
4. A window displays (Select users, computers...). In this window, select the **Administrators** Group.

   Note: This can be a bit ambiguous here, since you may see `Administrator` as a user, but you need to add the Administrator group, not the Administrator user.

   Click **Add** and then click **OK**.
5. In the Security tab, the Administrators Group has been added. You need to remove "`Everyone`". Select **Everyone** and clear the box that says "Allow inheritable permission...."
6. Click **Remove** on the Security window that is displayed.

Configuration Manager PDI Encrypt field

When configuring your WebSphere Commerce instance, it is recommended that you select the PDI Encrypt check box. Enabling the PDI Encrypt field specifies that information in the `ORDPAYINFO` and `ORDPAYMTHD` tables should be encrypted. By selecting the check box, payment information is stored in the WebSphere Commerce database in encrypted format.

Default Authentication Policies

WebSphere Commerce ships two default authentication policies:

- "Shoppers"
- "Administrators"

Shoppers

The default account policy for shoppers contains the default account lockout policy and default password policy for shoppers.

The default account lockout policy for shoppers contains the following default attributes:

Attribute	**Default value**
Account lockout threshold	6 attempts
Consecutive unsuccessful login delay	10 seconds

The default password policy for shoppers contains the following default attributes:

Attribute	**Default value**
Whether the user ID and password can match	N (no they cannot match)
Maximum occurrence of consecutive characters	3 characters
Maximum instances of any character	4 instances
Maximum lifetime of the passwords	180 days
Minimum number of alphabetic characters	1 alphabetic character
Minimum number of numeric character	1 numeric character
Minimum length of password	6 characters
Whether the user's previous password can be reused	N (no it cannot be reused)

Shoppers that perform a self-registration are assigned the default shopper authentication policy - Shoppers.

Administrators

The default account policy for administrators contains the default account lockout policy and default password policy for administrators.

The default account lockout policy for administrators contains the following default attributes:

Attribute	**Default value**
Account lockout threshold	3 attempts
Consecutive unsuccessful login delay	20 seconds

The default password policy for shoppers contains the following default attributes:

Attribute	Default value
Whether the user ID and password can match	N (no they cannot match)
Maximum occurrence of consecutive characters	3 characters
Maximum instances of any character	4 instances
Maximum lifetime of the passwords	90 days
Minimum number of alphabetic characters	1 alphabetic character
Minimum number of numeric character	1 numeric character
Minimum length of password	8 character
Whether the user's previous password can be reused	N (no it cannot be reused)

The default `wcsadmin` administrator user that is shipped with WebSphere Commerce is assigned the default authentication policy - `Administrators`.

Chapter 5. Session management

Web browsers and e-commerce sites use HTTP to communicate. Since HTTP is a stateless protocol (meaning that each command is executed independently without any knowledge of the commands that came before it), there must be a way to manage sessions between the browser side and the server side.

WebSphere Commerce supports two types of session management: cookie-based and URL rewriting. The administrator can choose to support either only cookie-based session management or both cookie-based and URL rewriting session management. If WebSphere Commerce only supports cookie-based, shoppers' browsers must be able to accept cookies. If both cookie-based and URL rewriting are selected, WebSphere Commerce first attempts to use cookies to manage sessions; if the shopper's browser is set to not accept cookies then URL rewriting is used.

Cookie-based session management

When cookie-based session management is used, a message (cookie) containing user's information is sent to the browser by the Web server. This cookie is sent back to the server when the user tries to access certain pages. By sending back the cookie, the server is able to identify the user and retrieves the user's session from the session database; thus, maintaining the user's session. A cookie-based session ends when the user logs off or closes the browser. Cookie-based session management is secure and has performance benefits. Cookie-based session management is secure because it uses an identification tag that only flows over SSL. Cookie-based session management offers significant performance benefits because the WebSphere Commerce caching mechanism only supports cookie-based sessions, and not URL rewriting. Cookie-based session management is recommended for shopper sessions.

If you are not using URL rewriting and you want to ensure that users have cookies enabled on their browsers, check **Cookie acceptance test** on the Session Management page of Configuration Manager. This informs the shopper that if their browser does not support cookies, or if they have turned off cookies, they need a browser that supports cookies to browse the WebSphere Commerce site.

For security reasons, cookie-based session management uses two types of cookies:

- Non-secure session cookie

  Used to manage session data. Contains the session ID, negotiated language, current store and the shoppers preferred currency when the cookie is constructed. This cookie can flow between the browser and server under either SSL or non-SSL connection. There are two types of non-secure session cookies:

  – A WebSphere Application Server session cookie is based on the servlet HTTP session standard. WebSphere Application Server cookies persist to memory or to the database in a multinode deployment. For more information, search for ″session management″ in the WebSphere Application Server Information Center (`http://www.ibm.com/software/webservers/appserv/infocenter.html`).

  – A WebSphere Commerce session cookie is internal to WebSphere Commerce and does not persist to the database.

To select which type of cookie to use, select `WCS` or `WAS` for the **Cookie session manager** parameter on the Session Management page of Configuration Manager.

- Secure authentication cookie

  Used to manage authentication data. An authentication cookie flow over SSL and is time stamped for maximum security. This is the cookie used to authenticate the user; whenever a sensitive command is executed, for example, the `DoPaymentCmd` which asks for a users credit card number. There is minimal risk that this cookie could be stolen and used by an unauthorized user. Authentication code cookies are always generated by WebSphere Commerce whenever cookie based session management is in use.

Both the session and authentication code cookies are required to view secure pages.

For cookie errors the `CookieErrorView` is called under the following circumstances:

- The user has logged in from another location with the same Logon ID.
- The cookie became corrupted, or was tampered with or both.
- If cookie acceptance is set to "`true`" and the user's browser does not support cookies.

Using cookies for session management

To use cookies in WebSphere Commerce, do the following:

1. Open Configuration Manager.
2. Select the **Instance**, then open the **Session Management** folder.
3. Select the appropriate session values.
   - Cookie acceptance test

     Select this check box to check if the customer's browser accepts cookies for a site that only supports cookies.
   - Cookie session manager

     Select whether you want WebSphere Commerce or WebSphere Application Server to manage your cookies. The default is WebSphere Commerce.
     - A WebSphere Application Server session cookie is based on the servlet HTTP session standard. WebSphere Application Server cookies persist to memory or to the database in a multinode deployment. For more information, search for "session management" in the WebSphere Application Server Information Center (`http://www.ibm.com/software/webservers/appserv/infocenter.html`.
     - A WebSphere Commerce session cookie is internal to WebSphere Commerce and does not persist to the database.
4. Click the **Advanced** Tab. Select the appropriate session values.
   - Cookie path

     Specifies the path for the cookie, which is the subset of URLs to which a cookie should be sent. Usually, this field should not be altered.

     For details on cookie paths, see Netscape's Cookie Specification and RFC 2109.
   - Cookie domain

     Specifies a domain restriction pattern. Usually, this field should not be altered.

     A domain specifies the servers that should see a cookie. By default, the cookies are only sent back to the WebSphere Commerce Server that issued them. By default, cookies are returned only to the host that saved them. Specifying a domain name pattern overrides this. The pattern must begin with a dot and must contain at least two dots. A pattern matches only one entry beyond the initial dot. For example, "`.ibm.com`" is valid and matches "`a.ibm.com`" and "`b.ibm.com`" but not "`www.a.ibm.com`". For details on domain patterns, see Netscape's Cookie Specification and RFC 2109.

5. Click **Apply**.
6. Close Configuration Manager.
7. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

URL rewriting

With URL rewriting, all links that are returned to the browser or that get redirected have the session ID appended to them. When the user clicks these links, the rewritten form of the URL is sent to the server as part of the client's request. The servlet engine recognizes the session ID in the URL and saves it for obtaining the proper object for this user. To use URL rewriting, HTML files (files with `.html` or `.htm` extensions) cannot be used for links. To use URL rewriting, JSP files must be used for display purposes. A session with URL rewriting expires when the shopper logs off.

Note: WebSphere Commerce dynamic caching and URL rewriting cannot interoperate. With URL rewriting turned on, you need to disable WebSphere Commerce dynamic caching. For more information, see the chapter on dynamic caching in the *WebSphere Commerce Administration Guide*.

Using URL rewriting session management

To specify how sessions should be managed, do the following:

1. Open Configuration Manager.
2. Select the **Instance**, then open the **Session Management** folder.
3. Select the appropriate session values.

   Enable URL rewriting. Select this check box to use URL rewriting for session management.

   Cookie session manager. Select WebSphere Application Server.
4. Click **Apply**.
5. Close Configuration Manager.
6. From the WebSphere Application Server Administration Console, stop then restart the WebSphere Commerce server instance.

Writing JSP templates for URL rewriting

If you want to use URL rewriting to maintain session state, do not include links to parts of your Web application in plain HTML files. This restriction is necessary because URL encoding cannot be used in plain HTML files. To maintain state using URL rewriting, every page that the user requests during the session must have code that can be understood by the Java interpreter. If you have such plain HTML files in your Web application and portions of the site that the user might access during the session, convert them to JSP files. This will impact the application writer because, unlike maintaining sessions with cookies, maintaining sessions with URL rewriting requires that each JSP template in the application must use URL encoding for every HREF attribute on <A> tags. Sessions will be lost if one or more JSP templates in an application do not call the encodeURL(String url) or encode RedirectURL(String url) methods.

Writing links

With URL rewriting, all links that you return to the browser or redirect must have the session ID appended to them. For example, this link in a Web page:

```
<a href="store/catalog">
```

is rewritten as

```
<a href="store/catalog;$jsessionid$DA32242SSGE2">
```

When the user clicks this link, the rewritten form of the URL is sent to the server as part of the client's request. The Servlet Engine recognizes ;$jsessionid$DA32242SSGE2 as the session ID and saves it for obtaining the proper HttpSession object for this user.

The following example shows how Java code may be embedded within a JSP file:

```
<%
 response.encodeURL ("/store/catalog");
%>
```

To rewrite the URLs you are returning to the browser, call the encodeURL() method in your JSP template before sending the URL to the output stream. For example, if a JSP template that does not use URL rewriting has:

```
out.println("<a href=\"/store/catalog\">catalog</a>")"
```

replace it with:

```
out.println("<a href=\"");
out.println(response.encodeURL ("/store/catalog"));
out.println("\">catalog</a>");
```

To rewrite the URLs you are redirecting, call the encodeRedirctURL() method. For example, if your JSP template has:

```
response.sendRedirect (response.encodeRedirectURL ("http://myhost/store/catalog"));
```

The encodeURL() and encodeRedirectURL() methods are part of the HttpServletResponse object. In both cases, these calls check to see if URL rewriting is configured before encoding the URL. If it is not configured, it returns the original URL.

Writing Forms: To write forms for submission, call the response.encodeURL("Logon"); on the ACTION tag of the form template. For example,

```
String strLoginPost = response.encodeURL("Logon");
<FORM NAME="Logon" METHOD="post" ACTION= <%= strLoginPost %> >
...
</FORM>
```

Writing the first page: The entry page, usually the home page, cannot contain frames. If you want to use frames in your store, you can have a non-frame page with a link to the store act as the store's entry page. However, if the store does use frames and a customer tries to access those pages with frames without going through the entry page first, their session may be lost. Customers can also lose their session if they use the **Back** button (only with frames) to return to the entry page and refresh the entry page. Refreshing the entry page gives them a new session ID. A link back to the entry page as an alternative to the **Back** button is necessary to help prevent this type of session loss.

Store-level session management

The diagram below illustrates the WebSphere Commerce store level registration infrastructure. Store level registration uses access control roles to associate a shopper with a store.

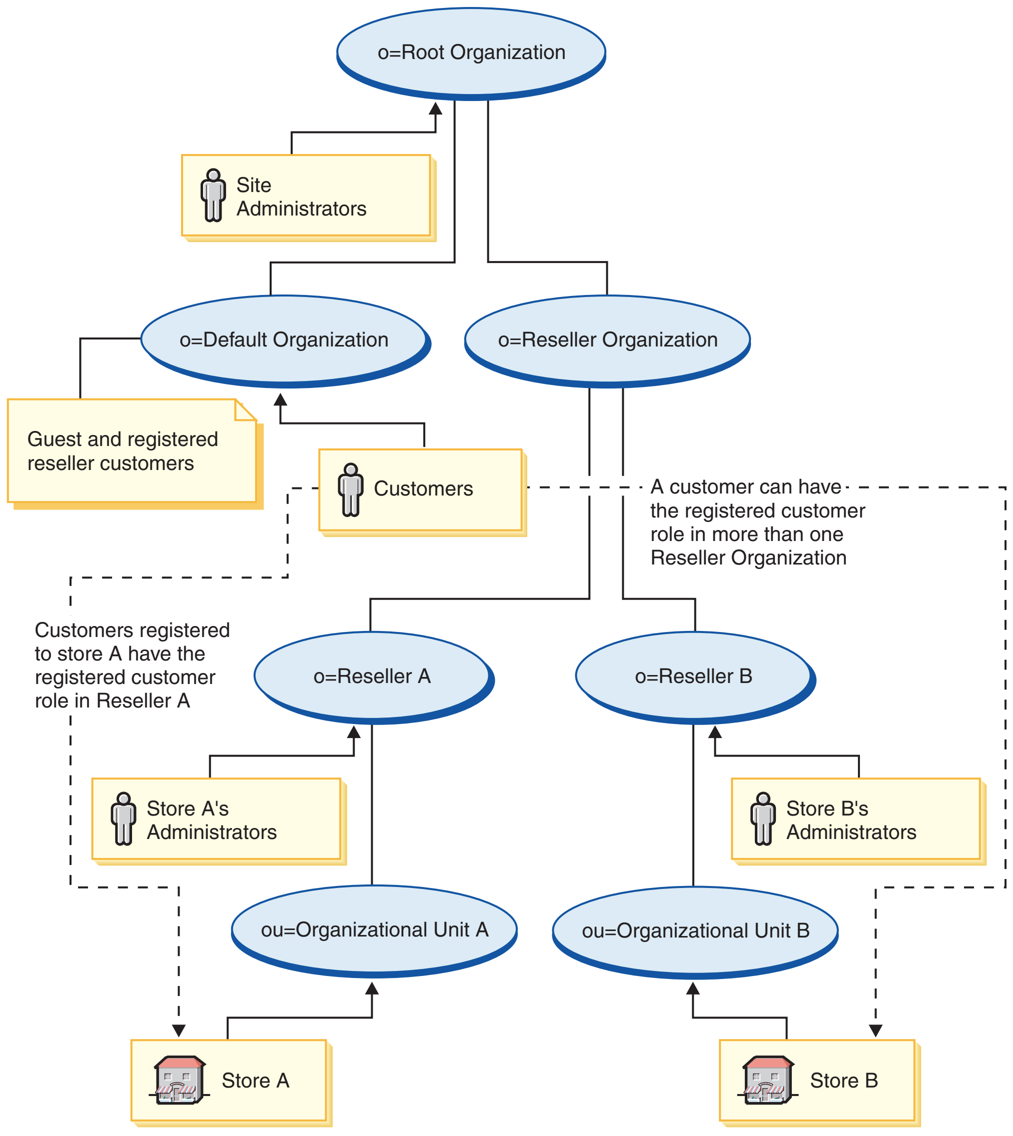


Figure 3. Store level registration

Users that shop at a store do not necessarily need to be a member of the store's organization but need to play a shopping role (that is, Registered Customer) in the organization. Users that play an administrative role in an organization are usually associated with the organization by having an ancestral relationship with the organization.

For example, suppose that you have a store, Store A as in the above diagram. Also suppose that Sue shops at Store A and Joe is an employee for Store A responsible for the administrative duties of running Store A. To model this scenario from an organizational perspective, Joe should be placed under the Store A's organization but Sue should not. As Sue is not an employee of Store A, Sue is associated with Store A by having her play the shopping role in the Store A organization.

A store determines all of its registered shoppers by finding all the users that play a shopping role in the store's organization. A user administrator of the store can then proceed to perform store wide activities such as setting up a campaign for all the registered users in a store, or specific actions such as resetting the password of a user registered to its store.

Referring to the diagram in Figure 3 on page 69, consider the following scenario:
- Sue, who is a member of the Default Organization, has a shopping role in Reseller A's organization. Reseller A's parent organization is the Reseller Organization.
- Reseller A owns store A
- Sue does not have an organizational role in Reseller B's organization
- Reseller B owns store B

Sue logs into Store A and shops as usual. When Sue accesses Store B, Sue is assigned a new session identity for Store B as a guest user. If she accesses Store A once again, the information in her previous session identity for Store A is used by WebSphere Commerce to manage her session.

The session identity for Store A would be reused for Store B if:
- Store A and Store B belong to the same organization.
- Sue has a role defined in both the Reseller A and Reseller B organizations.

Chapter 6. Setting and changing passwords

Most components in WebSphere Commerce utilize user IDs and passwords that are validated by the operating system. For information on changing those passwords, refer to your operating system documentation. This chapter covers how to set and change passwords for WebSphere Commerce components that do not validate user IDs and passwords through the operating system.

Quick reference to user IDs, passwords and Web addresses

Administration in the WebSphere Commerce environment requires a variety of user IDs. These user IDs along with their requisite authorities are described in the list below. For the WebSphere Commerce user IDs, the default passwords are identified.

400 **iSeries user profiles**

Two iSeries user profiles are used and referred to frequently when you install and configure WebSphere Commerce:

- A user profile which you create and use to install WebSphere Commerce and start the Configuration Manager. To install and configure WebSphere Commerce, you must use an iSeries user profile of `USRCLS(*SECOFR)` or use the `QSECOFR` user profile. If you need to create a user profile, refer to the*WebSphere Commerce Installation Guide* for iSeries.
- A user profile which is created by the Configuration Manager when you create a WebSphere Commerce instance. This user profile is also referred to as the *instance user profile*. A user profile of `USRCLS(*USER)` is created by the Configuration Manager each time you create a WebSphere Commerce instance. If you need to create a user profile, refer to the *WebSphere Commerce Installation Guide* for iSeries.

Configuration Manager user ID

The Configuration Manager tool's graphical interface allows you to modify the way WebSphere Commerce is configured. The default Configuration Manager user ID and password are `webadmin` and `webibm`.

AIX Linux Solaris Windows You can access Configuration Manager from your WebSphere Commerce machine, or any machine on the same network as WebSphere Commerce.

400 For iSeries, you can access Configuration Manager from any Windows machine that is on the same network as your iSeries server.

IBM HTTP Server User ID

AIX Linux Solaris Windows If you are using IBM HTTP Server, you can access your Web server home page by opening your Web browser and typing the following Web address:

http://*host_name*

If you have customized your Web server, you may be required to type the name of your Web server's front page after the host name.

WebSphere Commerce Instance Administrator

The Instance Administrator user ID and password apply to the following WebSphere Commerce tools:

- WebSphere Commerce Accelerator. To access the WebSphere Commerce Accelerator from a remote machine running a Windows operating system, open your Internet Explorer Web browser, and type the following Web address:

  `https://host_name:8000/accelerator`
- WebSphere Commerce Administration Console. To access the WebSphere Commerce Administration Console from a remote machine running a Windows operating system, open your Internet Explorer Web browser, and type the following Web address:

  `https://host_name:8002/adminconsole`
- WebSphere Commerce Organization Administration Console. To access the WebSphere Commerce Organization Administration Console from a remote machine running a Windows operating system, open your Internet Explorer Web browser, and type the following Web address:

  `https://host_name:8004/orgadminconsole`

For the above tools, enter the administrator user ID and password that you entered when you created your WebSphere Commerce instance.

Note: The site administrator user ID should never be removed, and should always have instance administrator authority.

WebSphere Commerce requires that the user ID and password adhere to the following rules:
- The password must be at least 8 characters in length.
- The password must include at least 1 numeric digit.
- The password does not contain more than 4 occurrences of a character.
- The password does not repeat the same character more than 3 times.

WebSphere Commerce Payments Administrator

When you install WebSphere Commerce Payments, the WebSphere Commerce Site Administrator ID is automatically assigned the Payments Administrator role. Follow the instructions in the *WebSphere Commerce Installation Guide* to switch the Payments Realm Class to WCSRealm if it has not already been done.

The Payments Administrator role enables a user ID to control and administer WebSphere Commerce Payments.

400 **Note:**
- Do not delete or rename the Site Administrator user ID created for your instance, and do not change any preassigned WebSphere Commerce Payments roles, as WebSphere Commerce functions related to WebSphere Commerce Payments integration will not work.

Windows Windows User ID

Your Windows user ID *must* have Administrator authority. If you are using DB2®, it requires that the user ID and password adhere to the following rules:
- They cannot be more than 8 characters in length.
- They can contain only the characters A to Z, a to z, 0 to 9, @, #, $, and _.
- They cannot begin with an underscore (_).
- The user ID cannot be any of the following, in upper, lower, or mixed case: USERS, ADMINS, GUESTS, PUBLIC, LOCAL.

- The user ID cannot begin with any of the following in upper, lower, or mixed case: IBM, SQL, SYS.
- The user ID cannot be the same as any Windows service name.
- The user ID must be defined on the local machine, and belong to the Local Administrator's group.
- The user ID must have the *Act as part of the operating system* advanced user right.

You can perform the installation without the *Act as part of the operating system* advanced user right, however, the DB2 setup program will be unable to validate the account that you specify for the Administration Server. It is recommended that any user account used to install DB2 has this advanced user right.

Important

If your Windows user ID does *not* have Administrator authority, is more than 8 characters in length, or is not defined on the local machine, you will be notified of the problem and will not be able to proceed with the installation.

If you are using DB2, you will use this user ID as the DB2 database user name (database user logon ID).

If you need to create a user ID fitting the above criteria, you can find information on creating a Windows user ID in the Windows online help.

Changing the Configuration Manager Password

You can change the Configuration Manager password when you launch the Configuration Manager by clicking **Modify** in the window where you enter your user ID and password.

Alternately, to change the Configuration Manager user ID or password switch to the `bin` subdirectory under the WebSphere Commerce installation path and type the following in a command window:

1. Change to the WebSphere Commerce `bin` subdirectory:

   ```
   cd  WC55_installdir/bin
   ```

2. Run the `wcs_encrypt` script to obtain an encrypted version of your password:

   AIX 400 Linux Solaris

   ```
   ./wcs_encrypt.sh new_password
   ```

   Windows

   ```
   wcs_encrypt new_password
   ```

3. Open the `PwdMgr.xml` file in the *`WC55_installdir`*`/instances` directory, and modify LoginPassword with the encrypted password encrypted in step 2.

Setting Your IBM HTTP Server Administrator Password

AIX Linux Solaris Windows To set your IBM HTTP Server administrator password,

1. Switch to the *HTTPServer_installdir*/`bin` directory on your machine.
2. Type the following command:

   AIX Linux Solaris `./htpasswd -b ../conf/admin.passwd` *`user password`*

   Windows `htpasswd -b conf\admin.passwd` *`user password`*where *user* and *password* are the user ID and password that you want to have administrative authority for IBM HTTP Server.

You have now successfully set your IBM HTTP Server administration password.

Note: If the administrator password does not exist, you need to run `htpassword` with the -c option to create the password first.

Changing Your SSL Key File Password

AIX Linux Solaris Windows If you are using IBM HTTP Server, follow the steps below to change your SSL key file password.

1. Windows Click **Start Menu** → **Programs** → **IBM HTTP Server** → **Start Key Management Utility**.
2. From the **Key Database File** menu, select **Open**.
3. Switch to the `ssl` subdirectory under the IBM HTTP Server installation path on your machine. Your key file (which has the file extension `.kdb`) should be in this folder. If not, create a new key file by following the instructions outlined in Chapter 17, "Enabling SSL for production with IBM HTTP Server," on page 185.
4. From the **Key Database File** menu, select **Change Password**. The Change Password window appears.
5. Enter your new password, and enable **Stash the password to a file**.
6. Click **OK**. Your password has been changed.

You have now successfully changed your SSL key file administration password.

Generating WebSphere Commerce encrypted passwords

You can generate encrypted passwords in order to manually reset the password of a user from a command line. There are other tools (such as the ResetPassword command) that accomplish the same task. To manually reset the password, the administrator would take the encrypted password that is output by the utilities below and update the `LOGONPASSWORD` field of the `USERREG` table. The administrator would also update the `SALT` field of the `USERREG` table with the chosen salt.

AIX Linux Solaris Windows WebSphere Commerce allows you to generate encrypted passwords. To generate encrypted passwords, do the following:

1. Go to the `bin` subdirectory under the WebSphere Commerce installation directory.
2. Run the following script from a command line:

   Windows `wcs_password.bat` *`password SALT merchant_key`*

AIX Linux Solaris `./wcs_password.sh` *`password SALT merchant_key`*

where

- *password* is the plain text password.
- *SALT* is a random string that is used in the generation of a password. This is found in the `SALT` column of the `USERREG` database table for the particular user whose password is being updated.
- *merchant_key* is the merchant key that was entered during instance creation.

400 For iSeries, to change the encrypted password for shoppers, use the `chgwcspwd.sh` command.

1. Start a QShell session on your iSeries system.
2. Navigate to the following directory: *`WC_installdir`*`/bin`
3. Run the following script from a command line: `chgwcspwd.sh` (The usage parameters will be displayed.)
4. Run the command again using the appropriate parameters.

For the details of running this command, see the WebSphere Commerce Production and Development online help.

Generating WebSphere Commerce Payments encrypted passwords

WebSphere Commerce allows you to generate encrypted passwords for WebSphere Commerce Payments. To generate encrypted passwords, do the following:

1. Go to the `bin` subdirectory under the WebSphere Commerce installation directory.
2. Run the following script from a command line:

   Windows `wcs_pmpassword.bat` *`password SALT`*

   AIX 400 Linux Solaris `./wcs_pmpassword.sh` *`password SALT`*

   where:

   - *password* is the plain text password.
   - *SALT* is a random string that is used in the generation of a password. This is found in the `SALT` column of the `USERREG` database table for the particular user whose password is being updated.

Resetting an administrator account

If a WebSphere Commerce account gets locked or disabled for some reason, you can unlock or enable it as follows:

If the account *is not* a site administrator's account:

1. Open the Administration Console.
2. Click **Access Management** > **Users**.
3. Double-click the user account or select the user account from the list and click **Change**.
4. Select **Enable** in the Account status field.
5. Click **OK**.

If the account *is* a site administrator's account or any other user account, run the following SQL statements from either a DB2 command window, or an SQLPlus prompt (for Oracle databases):

```
CONNECT TO db_name [USER user_id USING password]
UPDATE USERREG SET STATUS=1, PASSWORDRETRIES=0 WHERE LOGONID='logonId'
```

where

db_name
: Is your WebSphere Commerce database name (for example, MALL).

user_id Is the database administrator user ID for the database.

password
: Is the password corresponding to the database administrator user ID.

logonId
: Is the user ID of the account that you want to reset (for example, `wcsadmin`).

For example, to reset the `wcsadmin` account, you can issue the following SQL statements if you are logged onto your system as the database administrator user ID:

```
CONNECT TO mall
UPDATE USERREG SET STATUS=1, PASSWORDRETRIES=0 WHERE LOGONID='wcsadmin'
```

400 To enter SQL statements on the iSeries platform, you can use either the DB2/400 Query Manager and the SQL development kit, or you can use iSeries Navigator. To use IBM iSeries Access to perform database queries, do the following:

1. Start iSeries Navigator from the PC where it is installed.
2. Expand the iSeries system. Expand Databases, right-click the Relational Database, and select **Run SQL Scripts**. The Run SQL Scripts window opens.
3. From the Connection menu, select **JDBC Setup**. Click the **Server** tab.
4. In the Default libraries field, erase any existing values and enter the name of the database schema of your instance. By default the schema name is the name of the instance. Click **OK** to save your changes.
5. Type the above SQL statements in the window.

LDAP server password storage consideration

Since you have chosen to use LDAP as your user registry, you should decide how you want your passwords to be stored. Many LDAP servers will provide options for encrypting or hashing the passwords, and sometimes even allow the passwords to be stored in plain text. Although the decision on how to store the password has no impact on the integration with WebSphere Commerce, you should investigate your settings and ensure that they comply with your security requirements.

For information on configuring your LDAP server, refer to the product documentation provided with your LDAP server.

Chapter 7. Single sign-on

This chapter outlines how to set up single sign-on for WebSphere Commerce.

Prerequisites

To enable single sign-on, you must meet the following requirements:

- There must be an existing LDAP server installed and configured. To configure an LDAP server see the *WebSphere Commerce Additional Software Guide*.
- WebSphere Commerce must be installed and configured to use LDAP.
- WebSphere Application Server security must be enabled. To enable WebSphere Application Server security see Chapter 16, “Enabling WebSphere Application Server security,” on page 173.

Enabling single sign-on

Attention

There are several key limitations of single sign-on when it is used with WebSphere Commerce. These limitations are:

- The LTPA cookies may flow across different web server ports.
- You may need to modify the `ldapentry.xml` file and add the object class `ePerson`. That is as an attribute of `ldapocs` element.
- You need to modify the `instance.xml` and ensure that the `MigrateUsersFromWCSdb` flag is set to "`ON`".
- The machines participating in the single sign-on configuration must have their system clocks synchronized.
- Single sign-on is only supported between applications that can read and issue the WebSphere Application Server Light Weight Third Party Authentication (LTPA) token.

To enable single sign-on you must do the following:

1. Enable single sign-on within the WebSphere Application Server. For more information, search for "single sign-on" in the WebSphere Application Server Information Center (`http://www.ibm.com/software/webservers/appserv/infocenter.html`). Select **Single Sign-On: WebSphere Application Server** and complete the following sections:
   - **Configuring SSO for WebSphere Application Server**.
     - **Modify WebSphere Application Server security settings**.

       Note: The step that details how to fill in the LDAP fields can be safely ignored.
     - **Export the LTPA keys to a file**.
2. On your WebSphere Commerce machine, start the WebSphere Commerce Configuration Manager.
3. To configure the **Member Subsystem** node, do the following:

a. Under **WebSphere Commerce** expand *host_name* → **Instance List** → *instance_name* → **Instance Properties** → **Member Subsystem**.
b. In the **Authentication Mode** drop-down menu, select **LDAP**.
c. Enable the **Single sign-on** check box.
d. In the **Host** field, enter the fully qualified host name of your LDAP server.
e. Enter the administrator's distinguished name in the **Administrator Distinguished Name** field. This should be the same name that was used on your LDAP server.
f. In the **Administrator Password** field, enter the administrator's password. This should be the same password that was used on your LDAP server. Confirm the password in the **Confirm Password** field.
g. Complete each of the remaining fields.
h. Click **Apply**, then click **OK**.

4. Configure the roles that will be assigned to users coming in to the system from single sign-on (SSO). Every time a user connects to the system by SSO WebSphere Commerce will try to assign the roles from the `MemberRegistrationAttributes.xml` file with registration type = "SSO". Link to new section describing MRA.xml.
5. Restart the WebSphere Application Server.

Configure roles for SSO users

In WebSphere Commerce 5.5, security roles are assigned as part of the registration process. With single sign-on, the customer can bypass the registration step for your site if they have successfully authenticated to a collaborating system. The ability to be implicitly authenticated to a WebSphere Commerce 5.5 site has very little value if the user will simply end up being denied access to the facilities that they want to use, for example, shopping at a store.

Therefore, the same functionality of automated role assignment that happens with user registration also happens in the session management code. In this case your would configure the roles for SSO shoppers using the 'SSO' registration type. This way, when a customer authenticates onto the system, WebSphere Commerce 5.5 will automatically provide all of the roles that they should have for the site. Keep in mind that the SSO role assignment happens on a site level and not on a store level (as with the typical user registration). Therefore, you should ensure that the storeAncestor attribute specified is actually an ancestor of the site (store 0).

Example:

```
<User registrationType="SSO" memberAncestor="o=Default Organization,o=Root Organization" storeAncestor="o=Root Organization"><BR>
    <Role name="Registered Customer" roleContext="explicit" DN="o=Reseller Organization,o=Root Organization"/><BR>
    <Role name="Registered Customer" roleContext="explicit" DN="o=Seller Organization,o=Root Organization"/><BR>
    <Role name="Registered Customer" roleContext="explicit" DN="o=Supplier Organization,o=Root Organization"/><BR>
    <Role name="Registered Customer" roleContext="explicit" DN="ou=Supplier Hub Organization,o=Business Indirect Supplier Organization,
          o=Root Organization"/><BR>
  </User>
```

This example will give four roles to any shopper who comes in to the system from SSO

Chapter 8. Administering X.509 Certificates

WebSphere Commerce supports client certificate logon as a security mechanism, protecting both site and customer. The X.509 certificate supplements basic authentication for customers entering a site. A customer holding this certificate can access a secured WebSphere Commerce site, which has been enabled for client certificate authentication.

When creating a WebSphere Commerce instance, you select the Authentication Mode. The Authentication Mode is either Basic or X.509. The default is Basic authentication, which is logon authentication using a login ID and password. To activate logon authentication using X.509 certificates, select X.509 authentication.

Before you can begin using X.509 certificates, you must arrange for a trust relationship with an external certificate authority to handle electronic authentication of the X.509 certificates. If you are using Netscape Enterprise as your Web server, you will need to follow additional steps to enable the X.509 certificates on your Web server. Refer to the Netscape Enterprise Server documentation for more information and complete instructions.

X.509 users are accessible through the WebSphere Commerce Accelerator. Before X.509 certificate authentication is enabled, the administrator must ensure there is a client certificate, which is recognized by the server certificate and installed on the browser. Otherwise, the administrator will be unable to logon. When the administrator accesses the WebSphere Commerce Administration Console login window for the first time, a certificate shopper record is created and a shopper cookie is issued, similar to when a normal shopper accesses a secure URL. After the administrator logs on to the WebSphere Commerce Administration Console using the correct ID and password, an administrator cookie is issued, replacing the shopper cookie. An administrator will then have two user records: the administrator user and the previous shopper user.

An error message is displayed when:
- A user's X.509 certificate has been revoked by a site
- A client certificate does not contain the necessary information to guarantee that the shopper is unique in WebSphere Commerce.

The X.509 error view task is registered as `X509 ErrorView` in the `VIEWREG` database table.

Enabling X.509 certificates

When creating a WebSphere Commerce instance, you select either Basic authentication or X.509 authentication using the Configuration Manager. The default is Basic authentication, which is authentication using a logon ID and password.

To enable authentication using X.509 certificates, do the following:
1. Set up your IBM HTTP Web server SSL certificate. The SSL server certificate includes a list of client authorities for trust relationships. You may need to add additional client certificate authorities.
2. Start the WebSphere Commerce Configuration Manager.

3. Select **Instance Properties -> WebServer**.
4. Check the **X.509** box for Authentication Mode. Click **Apply**. X.509 client certificate users will now be accepted. The IBM HTTP server is automatically enabled for certificate support, when X.509 Authentication Mode is selected.
5. Start and stop the WebSphere Commerce server. WebSphere Commerce will not register X.509 users in the `CERT_X509` table until the server has been restarted.

Note: You can configure the IBM HTTP server to make X.509 certificates either optional or required.

1. Open the configuration file `httpd.conf` and locate the `SSLClientAuth` directive. Set the directive to 1 (optional) or 2 (required). The recommended parameter is *required*.
2. Since the WebSphere Commerce Payments client does not support SSL Client Authentication, you must disable SSL between the WebSphere Commerce Payments client and the Web server.
   a. In a text editor open the `PaymentServlet.properties` file. The file is in the WebSphere Commerce Payments installation Directory.
      - Locate the `UseNonSSLWCSClient` property. Set the property to a value of ’1’ (one).
      - If you cannot find the `UseNonSSLWCSClient` property in the file, add the line:

        ```
        UseNonSSLWCSClient=1
        ```
   b. Save the file, and exit the editor.
3. If WebSphere Commerce Payments is installed on the same machine as WebSphere Commerce:
   a. Start Configuration Manager.
   b. Select the instance; then select **Payments**.
   c. Check **Use non-SSL WebSphere Commerce Payments Client**. This enables the WebSphere Commerce Server client to communicate with WebSphere Commerce Payments, without using SSL.
   d. Click **Apply**.
   e. Close Configuration Manager.
4. Restart the WebSphere Commerce Payments application server from the WebSphere Administration Console.
5. Restart the WebSphere Commerce application server from the WebSphere Administration Console.

Refer to the IBM HTTP server documentation for more information and further options on setting restrictions and filtering parameters for certificates.

Updating the status of X.509 certificate users

Using the WebSphere Commerce Accelerator, a site administrator can update the status of an X.509 certificate user, to one of the three following status values:

Valid The user can access a secured WebSphere Commerce site with their certificate.

Revoked
The user cannot access the WebSphere Commerce site. When a revoked certificate user attempts to log on; they will get an X.509 certificate error page.

Expired
: The user cannot access the WebSphere Commerce site. When an expired certificate user attempts to log on; they will get an X.509 certificate error page.

When administering X.509 certificates, you may also wish to set restrictions and filtering parameters for certificate holders. For example, you may also want to allow certain types of certificate holders to access your secured site, by modifying the `httpd.conf` configuration file.

For more information and instructions, refer to your Web server documentation.

A typical authentication scenario

The following steps illustrate a typical authentication scenario for X.509 certificates:

1. A shopper accesses:
   - A non-secure URL through `http://`

     No authentication is performed.
   - A secure URL through `https://`

     The shopper is prompted to select a client certificate.
   - A URL command and is redirected to `https://` because of the access mode of the URL command

     The shopper is prompted to select a client certificate.
2. The WebSphere Commerce server uses the information from the client certificate to see if the shopper already exists in the WebSphere Commerce `SHOPPER` table:
   - If the shopper exists with a valid certificate status, the shopper is authenticated and the shopping flow resumes.
   - If the shopper does not exist:
     - The shopper is automatically registered in the WebSphere Commerce database and the shopping flow resumes.

Note: Only the information found in the `CERT_X509` table is taken from the certificate. However, shopper address information could be taken from the X.509 client certificate, if it is available .

Part 3. Administering security authorization

This part describes the security authorization tasks that can be performed by the WebSphere Commerce site administrator.

Chapter 9. An introduction to access control

The role of e-commerce has not only changed the way companies are doing business, but it has dramatically increased the kinds of relationships that they can expect to have with their customers and business partners. The Web is a key factor in delivering improved value to your existing customers, and paving the way for new customers eager to benefit from the power and increased efficiency of the Internet. Along with the clear advantages of doing business on the Web and the tremendous potential for increasing your customer base, comes the challenge of managing your business flows and trading patterns while maintaining a highly secure environment, authorizing appropriate transactions, and streamlining your work processes.

The hallmark of access control is the ability to oversee these work processes by managing the ways in which users participate in your system, based on their activities, and their business relationship to your products and services. For example, you might only want customers that have registered with your site to be able to view products for auctions in your store, and to place bids on them. Likewise, you might authorize graphic designers to customize your store pages, but you might restrict them from managing the actual content in your product catalog.

WebSphere Commerce provides you with the right tools for access management, by including more than 200 default access control policies that are automatically loaded into your system at the time of instance creation. These policies have been designed to address many of the typical access control requirements that your business needs, and can even be customized to suit your own e-commerce solution.

Managing access to activities in your electronic marketplace is an integral part of protecting your company’s financial assets and resources, for ensuring secure business transactions between approved members of your site, and validating the legitimacy of your online operations. Access control becomes especially crucial in the context of e-commerce, where the entry to your business is largely affected by customer relationships that begin over the Web.

What access control means for you

Access control enables you to manage your business work flows and ensure that users only carry out those activities that are appropriate with their roles and responsibilities. Not only does WebSphere Commerce provide you with default policies that are ready to use ″out of the box,″ but it also provides you with the tools and capacity to customize the policies to suit your business needs.

The following table outlines just a few examples of how simple modifications can customize access to your business environment.

What users are allowed to do by default	What users are allowed to do after customization
Customers can self-register.	Only seller administrators can register new customers.
Buyers can display RFQs that they created.	Only sellers can display RFQs if the RFQ resulted in a contract.

What users are allowed to do by default	What users are allowed to do after customization
Only customers can cancel orders they created if the order is in pending state.	Customer Service Representatives can also cancel orders in pending state, if the total product price is less than $1000.
An order can be modified by the person who created it.	Only a user from the buyer organization with the role of `purchaser` can modify an order that has been created.
Account representatives can display all accounts.	Account Representatives can only display active accounts.
Employees with the Logistics Manager role can create and modify fulfillment centers.	Employees with the Logistics Manager role can create but not modify fulfillment centers.

In the next chapter, we will take a look at how to create organizations and users, and at an access control policy in detail.

Chapter 10. Getting started

In the previous chapter, you learned about the important role that access control plays in e-commerce, and its key benefits for improving the efficiency and reliability of doing business over the Web.

In this chapter, we will discuss the basics of access management in WebSphere Commerce, such as defining organizations and users, and how access control policies are used to manage the activities that these organizations and their users perform across your system. After briefly outlining the steps you would take to set up your organizations and users, we will take a closer look at access control policies, their role in WebSphere Commerce, and study one in detail.

The chapter is divided into the following sections:
- Defining organizations and users
- Understanding access control
- How do I get started using access control?

Defining organizations and users

For site administrators, one of your first tasks after installing and configuring WebSphere Commerce is setting up and managing access to your e-commerce site. This involves creating organizations that will participate at your site, as well as defining the users that will be members of those organizations. In WebSphere Commerce 5.5, business models have been introduced. After an instance is created, there are sample business models that administrators can publish that will set up the organization structure. For more information on the business models, see "Business models" on page 17.

In some cases, the organizations joining your site may be buyer organizations, or else, you may have customers registering at your site who are engaged in a business-to-consumer relationship with your business. Regardless if you are administering a business-to-business or business-to-consumer site, defining the organizational structure of site is an important step in managing the types of access that members have to your system.

In this section, we will provide the high level steps that you need to take to define the structure of your site. If you are using the sample business models provided, you can skip ahead to the next section on access control. If you want to define your own organization structure, continue with the following steps.

To find details on creating organizations, users, and roles, refer to the online help, which is available from the Technical Library page:

Business

```
http://www.ibm.com/software/webservers/commerce/wc_be/lit-tech-general.html
```

Professional

```
http://www.ibm.com/software/webservers/commerce/wc_pe/lit-tech-general.html
```

We also recommend that you look at the *WebSphere Commerce Fundamentals*. For an overall look at the business models, see the *WebSphere Commerce Store Development Guide* , and *WebSphere Commerce Sample Store Guide* respectively.

Defining a seller organization

Typically, the seller organization is the organization that owns one or more stores on a WebSphere Commerce site. The seller organization can also have sub-organizations, or divisions, which in turn can have one or more stores. For example, the sample store, InFashion, which sells fashion merchandise, may have a women's division and a men's division, each with separate, online stores.

For now, let us assume that you are setting up a seller organization that does not have any sub-organizations. To set up the seller organization, here is a broad outline of what you will need to do:

1. Create a new organization. When you create a new organization, you will create a profile for that organization, which includes the organization's name, description, address, and contact person, as well as the organization type.
2. (Optional) Define which tasks within the seller organization require approval, such as order processing or user registration. This step is only required for a business-to-business site. Refer to the product online help for approvals documentation.
3. Assign roles to the new organization. An organization can only take on roles that have been assigned to its parent organization. Since Root Organization is an ancestor of all other organizations, it must be assigned all possible roles. WebSphere Commerce provides a set of default roles that you can start using right away. Since you are creating a seller organization, typical roles that you might assign include Seller Administrator, Seller, and so on. See "Roles" on page 28 for a list of default roles.
4. Create users. Like organizations, you will create a profile for each user that includes the user's name, contact information, and the role assigned to that user. When assigning roles, you will select them from the list of roles you assigned to the organization in the previous step.
5. Assign policy groups to the new organization so that customers can shop at the store that is managed by the organization. The typical policy groups required are: Management and administration policy group, common shopping policy group, B2C policy group, or B2B policy group. For more information on the policy groups, see "Default access control policies and groups," on page 201.

All of the steps outlined above can be done from the Access Management menu in the Organization Administration Console, by a Site Administrator.

Note: In WebSphere Commerce Professional Edition, you cannot create any organizations. A seller organization will already be created for you.

Defining a buyer organization

If you are running a business-to-business site, there can be one or more buyer organizations belonging to your site. (If you are running a business-to-consumer site, you will have individual buyers registered to the Default Organization instead). After you have established which businesses will participate in a buying relationship with your site, you will have to create a buyer organization for each business. You can have as many buyer organizations as you need.

Buyer organizations are structurally similar to seller organizations. Like seller organizations, buyer organizations can also have sub-organizations or divisions, that represent different buying activities for the organization.

For now, let us assume that your buyer organizations do not have any sub-organizations. To set up a buyer organization, here is an outline of what you need to do:

1. As you did when creating the seller organization, create a new organization and define approvable tasks if needed. Again, defining approvable tasks is only required for business-to-business sites.
2. Assign roles to the new buyer organization. Since you are now creating a buyer organization, typical roles that you might assign include Buyer Administrator, Buyer (buy-side), Buyer Approver, and so on.
3. Create users and assign them roles. When assigning roles, you will select them from the list of roles you assigned to the buyer organization in the previous step.
4. Repeat the entire procedure for each buyer organization you want to add to your site.

Note: Under normal circumstances, buyer organizations do not need to subscribe to any policy groups, since they will inherit the policy groups to which the Root Organization subscribes.

Again, all of the steps outlined above are done from the Access Management menu in the Organization Administration Console.

Note: In WebSphere Commerce Professional Edition, all customers belong to the Default Organization.

Understanding access control

After you have finished defining the organizations and users that will participate in your e-commerce site, you can now manage their activities through a set of policies, a process referred to as *access control*. In this next section, we will take a look at access control policies and their basic structure.

What is an access control policy?

An access control policy is a rule that describes which group of users is authorized to perform particular activities on your site. These activities can range from registration, to managing auctions, to updating the product catalog, and granting approvals on orders, as well as any of the hundreds of other activities that are required to operate and maintain an e-commerce site.

The policies are what grant users access to your site. Unless they are authorized to perform their responsibilities through one or more access control policies, users have no access to any of your site’s functions.

How does an access control policy work?

Access control policies consist of four parts: an access group, an action group, a resource group, and an optional relationship.

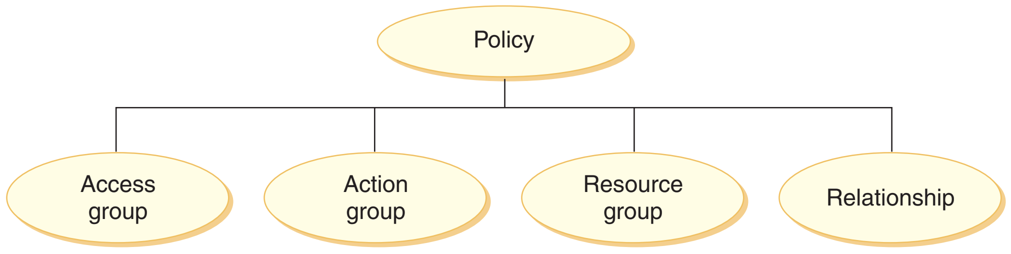
An *access group* is a group of users that share common access to a set of functions on your site. An access group generally includes users that share common attributes, such as the same role, department or skill set.

An *action group* refers to a group of actions that can be acted on the same resource. In general, action groups include actions that are associated with a common business area, or a related set of activities on your site.

A *resource group* includes the resources controlled by the policy. A resource group may include business objects such as a contract, or a set of related commands.

In some cases, a resource may only be acted on by a user that has a *relationship* to that resource. For example, only those users that create a contract might be allowed to modify it.

Figure 4. The four parts of an access control policy



Together, these four parts define a policy in WebSphere Commerce by specifying the users, the actions they can take, the business object or set of commands on which their actions are taken, and optionally, the relationship that the users have to the resource group.

For more detailed information on access groups, action groups, resource groups, and relationships, see Chapter 3, “Authorization concepts,” on page 17.

How do I get started using access control?

In some cases, you have to do nothing at all! The introduction of business models helps provide the basic access control structure in a system as well, the default policies in WebSphere Commerce are designed to provide a basic structure of access control based on typical users in your system, and the activities they perform that are associated with their roles in an organization. The policies cover a wide range of common business activities, including membership, order creation and processing, work flow approvals, and trading, such as auctions, request for quotes and contracts. After you have defined your organizations and users, the default policies can be used as provided or customized to meet your company’s individual needs.

However, before you can decide whether you want to use the default policies, or customize them, it is important understand what they look like in WebSphere Commerce. For a closer look at a default policy in detail, see “Looking at a policy in detail” on page 41.

Chapter 11. Customizing default access control policies

The default access control policies provided by WebSphere Commerce address the basic requirements that organizations have for regulating the actions and information available to their users. Often, the default policies may be sufficient for your site's needs. At the same time, the default policies are highly customizable, which enables you to tailor them to your own requirements.

The `SiteAdministratorsCanDoEverything` policy, is a special default policy that grants super-user access to administrators with the Site Administrator role. In this policy, a Site Administrator can perform any action on any resource, even if those actions or resources have not been defined. It is important to be aware of this when assigning this role to users.

This chapter provides information on how to make basic changes to the default access control policies included with WebSphere Commerce. We begin by introducing certain concepts and relationships you'll need to understand.

Note: If you encounter terms or concepts that are unfamiliar to you, see Chapter 3, "Authorization concepts," on page 17 for more information.

Identifying the policies affected by a change

In Chapter 3, "Authorization concepts," on page 17, you learned that policies are often related to other policies. You also learned how to start with a resource-level policy and identify the role-based policies associated with it. In this section we will explain in more detail how policies are related to each other and why you need to understand their relationships before you can modify an existing policy, or create a new one. In many cases, you need to change several policies to properly implement a change.

Understanding the relationship between role-based and resource-level policies

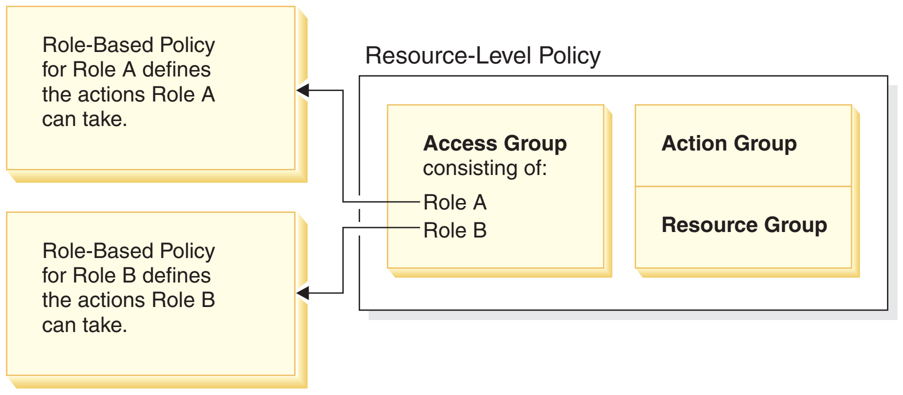
In WebSphere Commerce, each action that can be taken by a user is assigned to one or more roles using role-based policies as follows:

- Each default role has a corresponding access group. For instance, the access group for the role Seller is `Sellers`.
- Each "role-based" access group generally has two associated role-based policies:
  - A policy that defines the controller commands the role is authorized to execute.
  - A policy that defines the view actions the role is authorized to execute. View actions map to views in the `VIEWREG` table. For instance,`OperationalReportsHomeRHSView` displays a web page with the list of operational reports that the Seller has access to.

Some controller commands have only a role-based policy, but no resource-level policy. This occurs if the command is not operating on any protectable resources. For example, the command `SetCurrencyPreferenceCmd` does not need a resource-level policy since it can only change the currency preference for the user running the command. If it was able to change the currency preference of another user, then the user object would have to be protected, and a resource-level policy would be needed. .

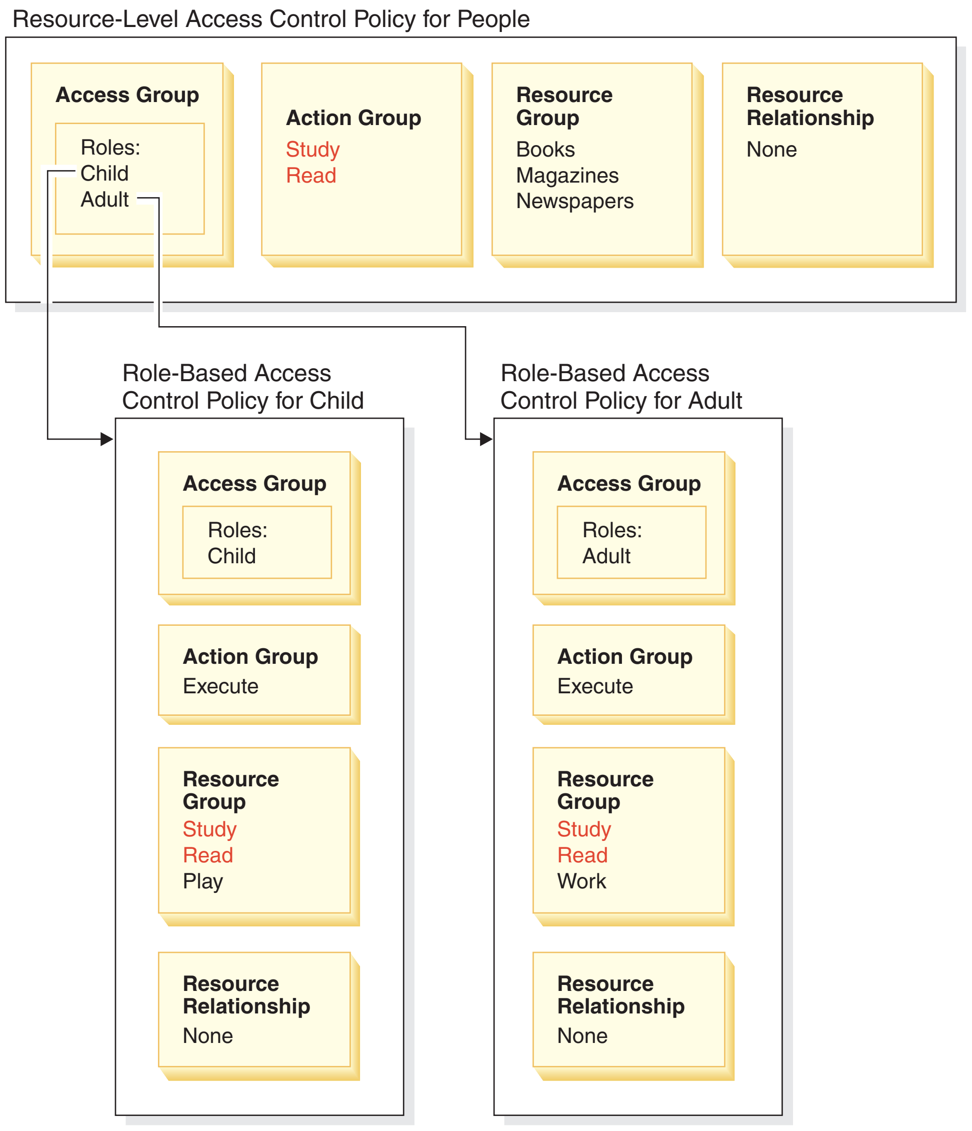
Resource-level policies for controller commands are directly related to certain role-based policies for controller commands. In the resource-level policy, the controller command is part of the action group, but in the role-based policy, the controller command is part of the resource group. The figure below illustrates this relationship. The resource-level policy includes Roles A and B in its access group, which brings the role-based policies for Roles A and B into play. While the resource-level policy grants authorization for users with roles A or B to take certain actions on a specific set of resources, the associated role-based policies provide authorization for users with roles A and B to take those actions in general.

Figure 5. Relationship between a resource-level policy and its associated role-based policies


Role-Based Policy for Role A defines the actions Role A can take.
Role-Based Policy for Role B defines the actions Role B can take.
Resource-Level Policy
Access Group consisting of:
Role A
Role B
Action Group
Resource Group

The following figure shows a sample resource-level policy that authorizes users in the `People` access group to read or study certain resources - namely books, magazines, and newspapers. This policy is correctly formulated because the role-based policies for the roles, `child` and `adult` also authorize them to read or study books, magazines, and newspapers.

Figure 6. A resource-level policy and the role-based policies that affect it.



Notice that in role-based policies for controller commands:

- The action group contains only a single action: `Execute`.
- The resource group contains the controller command that can be executed.

Similarly, in role-based policies for views:

- The action group contains the views that can be executed.
- The resource group contains a single resource: `com.ibm.commerce.command.ViewCommand`.

On the other hand, in resource-level policies:

- The action group contains the set of actions that can be performed on the resources in the resource group.
- The resource group contains a list of the actual business resources that can be acted upon.

A resource-level policy can only authorize users in a particular role to take actions already authorized by the corresponding role-based policy. For instance, in the above example, the role `child` is authorized to take the following actions:

- Study
- Read
- Play

Suppose that the resource-level policy is now changed to include a new action called `work`. Users with the role `adult` will be able to perform the action `work`. However, users with the role `child` will not. The reason for this is apparent when you check the role-based policies for the two roles. The policy for `adult` lists the action `work` in its resource group. The policy for `child` does not. Even though both `child` and `adult` are properly authorized by the resource-level policy, the role-based policy for `child` does not authorize the action `work`.

Because of the way resource-level policies are tied to role-based policies, the best way to track down all the policies affected by a particular change is to work backwards from the resource-level policy. The first step is to examine the access group of the resource-level policy and determine if it contains any roles. You can view the complete list of default roles by selecting Access Management > Roles from the Organization Administration Console.

If the resource-level policy's access group includes roles, review their role-based policies to see whether they need to be changed. If you are adding an action to the action group of a resource-level policy, you need to make sure that the relevant role-based policies also authorize the new action. If you are deleting an action from a resource-level policy, and no other resource-level policies reference this action, it's best to remove the corresponding resource from the associated role-based policies.

Understanding the policy model

An authorizing policy must be present for a user to perform an action. However, WebSphere Commerce permits users to take an action if **any** policy provides the needed authorization. Therefore, if you define a new, more restrictive policy than the default, you must delete or modify the broader default policy to prevent it from overriding your new policy.

For instance, suppose default policy A authorizes all registered users to submit auction bids. You want to change this policy so that auction bidding is limited to users with the buyer role. If you merely define a new policy that authorizes buyers to create auction bids, then your new policy will have no effect. Default policy A will still permit all registered users to bid. To make your new policy take effect, you must delete the broader, default policy.

The following table summarizes the additional changes you need to make when you create, delete, or change a resource-level policy.

Table 9. Additional changes needed when you change a resource-level policy that uses roles.

Change to a resource-level policy	**Required change if the resource-level's access group uses roles**
Add an action to the policy's action group.	Ensure that the applicable role-based policies include the action in their resource groups.

Table 9. Additional changes needed when you change a resource-level policy that uses roles. (continued)

Change to a resource-level policy	**Required change if the resource-level's access group uses roles**
Remove an action from the policy's action group.	No additional change required. For consistency, it is better to remove this action from the corresponding resource groups in the related role-based policies. This should only be done if no other action groups are referring to this action. If other action groups are referring to this action, likely there are role-based policies that still need to have this action in their resource group.
Use a different action group.	Ensure that the applicable role-based policies include in their resource groups the new action group's actions.
Add a role to the policy's access group.	Ensure that the role-based policy corresponding to the new role, refers to a resource group that includes the actions specified in the resource-level policy.
Remove a role from the policy's access group.	No additional change required. For consistency, it is better to modify the corresponding role-based policy so that it no longer references these actions in its resource group.
Use a different access group.	Ensure that the applicable role-based policies include in their resource groups the actions in the resource-level policy's action group.
Create a new policy.	Check whether there is an existing policy that authorizes the same actions. Delete if necessary.
Delete the policy.	To prevent any users from taking that policy's actions, delete any other policies that authorize the same actions.

Determining whether a policy is role-based or resource-level

Role-based policies are also known as command-level policies because they authorize users with a particular role to execute a set of commands. Resource-level policies authorize a group of users to execute a set of commands on a particular set of resources. For instance, a role-based policy might authorize children to eat. While a resource-level policy might authorize children eat rice.

You can usually determine whether a policy is a role-based policy or a resource-level policy by looking at its name.

Role-based policies

Policies that define the controller commands that a role can execute follow the naming convention:

<AccessGroupforRoleXYZ> Execute <XYZCmdResourceGroup>

For instance: `ProductManagersExecuteProductManagersCmdResourceGroup`.

In role-based policies for controller commands, the action group contains a single entry called `Execute` and the resource group contains a list of WebSphere Commerce commands that users with that role can execute.

Policies that define the views that a role can execute follow the naming convention:

<AccessGroupforRoleXYZ> Execute <XYZViews>

For instance: `SalesManagersExecuteSalesManagersViews`.

The resource group contains a single resource called `com.ibm.commerce.command.ViewCommand`.

Resource-level policies

Policies that define who can take actions on data resources (business objects that can be created or manipulated) follow the naming convention:

<AccessGroupXYZ> Execute <XYZCommands> On <XYZResource>

For instance: `AllUsersExecuteOrderProcessOnOrderResource`.

In resource-level policies, the action group contains WebSphere Commerce commands and the resource group identifies the specific business resources that can be acted upon.

One exception is policies that authorize the creation of an entity such as an order, a bid, or an RFQ. These policies do not act on the entity itself because it has not yet been created. Instead, they act on the containing entity. For instance, an auction is created in the context of a store, a user is created in the context of an organization. Most resources are created in the context of a store. Consequently, these policies have names such as:

<AccessGroupXYZs> Execute <XYZCommands> On <StoreEntityResource>

For instance:
`AuctionAdministratorsForOrgExecuteAuctionCreateCommandsOnStoreEntityResource`

.

Policies that define who can view DataBean resources (Data beans contain information about data resources such as a bid or an order; usually used in JSPs) follow the naming convention:

<AccessGroupXYZs> Display <XYZDatabeanResourceGroup>

For instance: `MembershipViewersForOrgDisplayMembershipDatabeanResourceGroup`.

Tips for changing default policies

Keep the following in mind when you change your default policies:

- Most access groups are defined by user roles such as buyer or product manager. To better understand these roles and what actions they are permitted to take, see “Roles” on page 28.
- Before you change a policy to use a different access group, review the definition of that access group to ensure it meets your requirements. To do so, select **Access Management > Access Groups** from the Organization Administration Console.
- Depending on the value you select for View, the Policies page lists the policies that are owned by the selected organization. It does not distinguish between site-level policies and policies specific to a particular organization.
- Rename any default policies you change so that the policy name reflects what the policy does and so that you can identify the default policies you have

changed. Consider implementing a naming convention for your customized policies. If appropriate, you should also modify the description of the policy and its display name.

Note: The access control policy menu is moved to Organization Administration Console. The Organization Administration Console can only perform simple modifications to the access control policy definitions and access group definitions. The more robust solution is to update the data using XML files. The following operations can only be done through XML:

1. Defining new actions, resources, attributes, relationships, relationship groups.
2. Defining complex implicit resource groups, and complex implicit access groups.
3. Assigning a new policy to a policy group.

After you make your policy changes

After a new policy has been created, the new policy must be assigned into a policy group before it comes into effect. You should assign the new policy to the group that serves the purpose of the policy. For more information on the policy group names, see "Default access control policies and groups," on page 201.

Each time you create or modify an access control policy, you must perform certain tests to verify that the policy is working correctly. Once you have finished testing all your new and changed policies that are currently in the database, it is a good idea to extract that information into XML files. These files have the same format as the initial access control policy related files: `defaultAccessControlPolices.xml`, `defaultAccessControlPolicies_locale.xml`, and `ACUserGroup_locale.xml`. This step is necessary because changes made using the Administration Console affect only the policy information stored in the database. The XML files that were used to load the default access control policies and their components during instance creation, are not updated automatically.

You should maintain consistency between the XML files and the access control information in the database for several reasons:

- When you create an instance of WebSphere Commerce, the policy and access group definitions are loaded from the XML files.
- The XML files offer a convenient way to directly view and edit your policies and their component parts so keeping the files up-to-date is essential.

Testing your policy changes

For each policy, ensure the following:

- A user that belongs to the policy's access group is able to take the specified actions on the specified resources. If you have removed authorization to perform an action, you should also test to make sure that the user can no longer perform the action.
- A user that does not belong to the policy's access group is unable to take the specified actions on the specified resources.

For example, suppose you implement Auction customization scenario 1 in Chapter 5, in which you remove the ability of auction administrators to close auction bidding. To test whether this change is working properly, log in as a user who belongs to the `auction administrator` access group and perform the following actions:

- Modify an auction
- Delete an auction.

You should also verify that an Auction Administrator cannot close bidding.

Then, log in as a user who does not belong to the `auction administrator` access group and attempt to perform the same actions. If the policy is working correctly, your attempts should fail.

Extracting your policy changes into the XML files

When you have finalized and tested your policy changes, you should update the XML files to keep them in sync with policy information in the databases. For a description of the different XML files related to access control policies and access groups, see Chapter 13, “Customizing access control polices using XML,” on page 129. Explanations on how to extract policy changes from the databases into the XML files and how to load the policy information from the XML files into the databases is also included.

Chapter 12. Customizing access control policies using the GUI

The scenarios presented below let you apply what you have learned about access control policies to make a variety of basic changes to your default policies using the GUI. If you want to make sophisticated changes, you will have to use the XML. See Chapter 13, “Customizing access control polices using XML,” on page 129.

For all of these scenarios, it is assumed that a Site Administrator is modifying the policies for Root Organization. Once you step through some of the scenarios, you will be able to follow the same methodology to make changes not specifically covered here.

The scenarios are organized by business area. Within each business area, the scenarios are presented in order of increased complexity.

Table 10. Table of contents for scenarios

Business area	Starting on page
Auctions	“Auctions scenario 1: Removing the ability of auction administrators to close auction bidding” on page 100
Contracts	“Contracts scenario 1: Remove the ability of contract managers to add or delete attachments to contracts” on page 103
Orders	“Orders scenario 1: Permitting only Buyers to create orders” on page 106
Membership	“Membership scenario 1: Remove the ability of users to self-register” on page 111
Coupons	“Coupons scenario 1: Allowing only buyers to redeem coupons” on page 116
Procurement	“Procurement scenario 1: Allowing procurement shopping cart managers to manage the procurement shopping cart for orders created by their organization” on page 120
Inventory	“Inventory scenario 1: Permit fulfillment center managers to update fulfillment centers but not to delete them” on page 123
Business intelligence	“Business intelligence scenario 1: Allowing auditors to view business intelligence reports” on page 124

If you are looking for a scenario that illustrates a particular kind of change, refer to Table 11, which cross-references the scenarios by the type of illustrated customization.

Table 11. Customization scenarios organized by type of customization

Customization	See page
Adding a role to a policy’s access group	118

Table 11. Customization scenarios organized by type of customization (continued)

Customization	See page
Changing a policy’s action group	121,123
Changing a policy’s resource relationship	108,120
Changing a policy to use a different access group	102,106,108,112,116,118
Creating a new access group and using it in a policy	110,113
Creating a new action group and using it in a policy	113,121
Creating a new resource-level policy	105,121
Creating a new role-based policy	113,124
Creating a new role and using it in a resource-level policy	113,124
Deleting a policy	101,112
Removing an action from a policy’s action group	3,103

Auctions scenario 1: Removing the ability of auction administrators to close auction bidding

By default, auction administrators for a store can modify or delete auctions for the store, as well as close bidding. In certain cases, you may not want to grant auction administrators the authority to close bidding, either because you want this action handled by others or because you do not require this action for your store.

In this scenario, you will remove the authority of auction administrators to close bidding. To accomplish this change, you will do the following:

1. Use the Appendix to find the resource-level policy that defines the actions that auction administrators can take.
2. Determine the name of the action group for the policy.
3. Delete the action for closing auction bidding from the policy’s action group.

Steps to take

Identify the policy whose action group must be changed

1. Look under Auctions, in the Appendix , to identify the resource-level policy to be changed. The policy is:

   `AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. Locate the policy in the list.
5. Note the name of the policy’s action group— `AuctionManage`. This is the action group you need to change to remove the action for closing bidding.

Remove the action for closing bidding from the policy's action group

1. Click **Access Management > Action Group**.
2. From the list of action groups, select **AuctionManage**.
3. Click **Change** to display the Change Action Group page.
4. From the Selected Actions list, select **com.ibm.commerce.negotiation.commands.CloseBiddingCmd**.
5. Click **Remove**.
6. Click **OK**.

Update the policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies** .
4. Click **Update**.

Auctions scenario 2: Removing the ability of auction managers to retract bids

By default, auction managers for a store can retract bids submitted at their auctions. In some cases, you might not want to grant this authority to anyone. To make this change, you must find the resource-level policy that defines who can retract bids and delete it.

In Auctions Scenario 1, the action, close bidding, was one of several included in the policy. Consequently, you had only to remove the action from the policy's action group. In this scenario, however, an entire policy controls bid retraction. Therefore, you must delete a policy not just an action.

To delete the policy, you need to do the following:

- Use the Appendix to find the resource-level policy that covers the retraction of auction bids by auction managers.
- Delete the policy.

Note: Before you delete the policy, make note of its name, access group name, resource group name, and action group name so you can recreate it for the next scenario.

Steps to take

1. Look under Auctions in the Appendix, to identify the resource-level policy to be changed. The policy is:
   `AuctionManagersForOrgExecuteAdminRetractBidCommandsOnAuctionResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. From the list of policies, select the following:
   `AuctionManagersForOrgExecuteAdminRetractBidCommandsOnAuctionResource`
5. Click **Delete**.

Update the policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.
5. Repeat steps 3 and 4 for the **Access Control Policy Groups Registry**.

Auctions scenario 3: Limiting auction bidding to buyers

By default, all registered users are permitted to bid for products being auctioned at a store, regardless of their position in their organization. In some cases, you may want to limit bidding to a restricted group of users such as those assigned the `buyer` role in WebSphere Commerce.

In this scenario, you will change a resource-level policy, as well as its associated role-based policy. To limit bidding to members of a buying organization with the `buyer` role, you need to do the following:

- Use the Appendix to find the resource-level policy that specifies who can create an auction bid.
- Change the policy's access group from all registered users, to those with the `buyer` role.
- Rename the policy, description, and display name.
- Identify the command for creating bids.
- Use the Appendix to find the role-based policy for buyers (buy-side). This policy defines the commands that users with the Buyer (buy-side) role can execute. You must update this policy's resource group to permit buyers to execute the command for creating bids.
- Update this role-based policy's resource group to include the command for creating bids.

Steps to take

Identify the resource-level policy

1. Look under Auctions, in the Appendix, to identify the resource-level policy to be changed. The policy is: `RegisteredApprovedUsersExecuteBidCreateCommandsOnAuctionResource`.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies it owns.
4. From the list of policies, select **RegisteredApprovedUsersExecuteBidCreateCommandsOnAuctionResource**.
5. Note the name of the policy's action group— `BidCreate`. This is the action group you need to view to find the name of the command for creating a bid.

Change the access group for the policy

1. Click **Change** to display the Change Policy page.
2. For User Group, click **Find** and select **Buyers (buy-side)**.
3. Click **OK**.
4. Rename the policy, display name, and description of the policy, by editing their text.
5. Click **OK**.

Identify the command for creating bids

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **BidCreate**.
3. Click **Change** to display the Change Action Group page. Note the name of the command for creating bids: `com.ibm.commerce.negotiation.commands.BidSubmitCmd`. You must add this command to the resource group that contains the list of commands a buyer can execute.

Identify the role-based policy and resource group for the Buyer (buy-side) role

1. Look under Role-Based Policies in the Appendix to find the role-based policy for buyers (buy-side). The policy is:
   `Buyers(buy-side)ExecuteBuyers(buy-side)CommandsResourceGroup`.
2. Click **Access Management** > **Policies**.
3. For View, select **Root Organization** to display the site-level policies.
4. Note the name of the resource group: `Buyers(buy-side)CommandsResourceGroup`. Now you have the name of the resource group you need to update.

Update the resource group in the role-based policy to include the command for creating bids

1. Click **Access Management > Resource Groups**.
2. Select **Buyers(buy-side)CommandsResourceGroup**.
3. Click **Change** to display the Change Resource Group page.
4. Click **Next** to display the Details page.
5. From the Available Resources list, select **com.ibm.commerce.negotiation.commands.BidSubmitCmd**. This is the command for creating bids.
6. Click **Add** to add the command to the resource group.
7. Click **Finish**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration >Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Contracts scenario 1: Remove the ability of contract managers to add or delete attachments to contracts

By default, contract managers for a store can add or delete attachments to contracts they manage. In some cases, you might not want to grant this authority to contract managers.

In this scenario, you will change a resource-level policy that defines the actions that a contract manager can take. To remove the authority of contract managers to add or delete attachments to contracts, you need to do the following:

- Use the Appendix to find the resource-level policy that defines the actions that contract managers can take.
- Determine the name of the action group for the policy.

- Delete the actions for adding attachments and deleting attachments from the list of actions in the policy's action group.

Steps to take

Identify the resource-level policy and action group

1. Look under Contracts, in the Appendix, to identify the resource-level policy to be changed. The policy is:

   `ContractManagersForOrgExecuteContractManageCommandsOnContractResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`ContractManage`. This is the action group you need to change to remove the actions for adding and deleting attachments.

Remove the actions for adding and deleting attachments from the policy's action group

1. Click **Access Management** > **Action Group**.
2. From the list of action groups, select **ContractManage**.
3. Click **Change** to display the Change Resource Group page.
4. From the Selected Actions list, select the following actions: **com.ibm.commerce.contract.commands.ContractAttachmentAddCmd com.ibm.commerce.contract.commands.ContractAttachmentDeleteCmd**.
5. Click **Remove**.
6. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Contracts scenario 2: Permit both contract operators and contract administrators to deploy contracts

By default, contract operators for a store can deploy contracts. In some cases, you might want to grant this authority to contract administrators as well.

The flexible design of access control policies offers several methods for implementing this change:

- You can create a new access group containing both contract operators and contract administrators and assign the new access group to the policy that defines who can deploy contracts.
- You can add the `deploy contract` actions to the policy that specifies the actions a contract administrator can perform.
- You can create a new policy that permits contract administrators to deploy contracts.

This scenario illustrates the third approach. It shows you how to create a new resource-level policy that authorizes contract administrators to deploy contracts.

To create this policy, you need to do the following:
- Use the Appendix to find the resource-level policy that authorizes contract operators to deploy contracts.
- Note the name of the action group for this policy.
- Note the name of the resource group for this policy.
- Define a new policy for the `contract administrator` access group, specifying the action group and resource group from the policy that authorizes contract operators to deploy contracts.

Steps to take

Identify the action group and resource group to use in the new policy

1. Look under Contracts, in the Appendix, to find the resource-level policy that authorizes contract operators to deploy contracts The policy is: `ContractOperatorsForOrgExecuteContractDeployCommandsOnContractResource`.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`ContractDeploy`. This is the action group you need to use in defining your new policy.
6. Note the name of the resource group—`ContractDataResourceGroup`, This is the resource group you need to use in defining your new policy.

Define the new policy

1. Click **New** to display the New Policy page.
2. For Name, specify:
   `ContractAdministratorsForOrgExecuteContractDeployCommandsOnContractResource`
3. For Display Name, specify a short description of the policy in your local language.
4. For Description, specify a longer description of what the policy does, in your local language.
5. For User Group, click **Find** and select **ContractAdministratorForOrg**.
6. Click **OK**.
7. For Resource Group, select **ContractDataResourceGroup**.
8. For Action Group, select **ContractDeploy**.
9. For Policy Type, select **Groupable Template Policy** to designate the policy as a template policy.
10. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Note: This new policy must be assigned to a policy group before it takes effect. The policy assignment must be done through XML. See for more information.

Orders scenario 1: Permitting only Buyers to create orders

By default, all users are permitted to create orders for products, regardless of their position in their organization. In some cases, you may want to limit the ability to create orders to a restricted group of users, such as the employees of the buying organization. Typically, these employees are assigned the Buyer (buy-side) role for the buying organization.

To limit order creation to users with the `Buyer` role, you need to do the following:

- Use the Appendix to find the resource-level policy that specifies who can create an order.
- Change the policy's access group from all users to those with the `Buyer` role.
- Update the policy's name, display name, and description.
- Identify the command for creating orders.
- Use the Appendix to find the role-based policy for Buyer (buy-side). This policy defines the commands that users with the Buyer (buy-side) role can execute. You must update this policy's resource group to permit buyers to execute the command for creating orders.
- Update this role-based policy's resource group to include the commands for creating orders.

Steps to take

Identify the resource-level policy

1. Look under Orders, in the Appendix, to identify the resource-level policy to be changed. The policy is: `AllUsersExecuteOrderCreateCommandsOnStoreResource`.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. From the list of policies, select **AllUsersExecuteOrderCreateCommandsOnStoreResource.** Note the name of the policy's action group—`OrderCreateCommands`. This is the action group you need to view to find the names of the commands for creating an order.

Change the access group

1. Click **Change** to display the Change Policy page.
2. For User Group, click **Find** and select **Buyers (buy-side)**.
3. Click **OK**.
4. Update the policy's name, display name, and description to reflect the change of access group.
5. Click **OK**.

Identify the command for creating orders

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **OrderCreateCommands** .
3. Click **Change** to display the Change Action Group page. Note the names of the commands for creating orders:

```
com.ibm.commerce.order.commands.OrderCopyCmd
com.ibm.commerce.order.commands.OrderScheduleCmd
com.ibm.commerce.orderitems.commands.OrderItemMoveCmd
com.ibm.commerce.orderitems.commands.OrderItemUpdateCmd
com.ibm.commerce.requisitionlist.commands.RequisitionListSubmitCmd
com.ibm.commerce.orderitems.commands.OrderItemAddCmd
com.ibm.commerce.orderquotation.commands.OrderQuotationCreateCmd
```

You must add these commands to the resource group that contains the list of commands a buyer can execute.

Note: The command, `com.ibm.commerce.orderitems.commands.AdminOrderItemUpdateCmd`, is not needed.

Identify the role-based policy for buyers (buy-side)

1. Look under Role-Based Policies, in the Appendix, to find the role-based policy for buyers (buy-side). The policy is: `Buyers(buy-side)ExecuteBuyers(buy-side)CommandsResourceGroup`.
2. Click **Access Management** > **Policies**.
3. For View, select **Root Organization** to display the site-level policies.
4. Locate the policy in the list.
5. Note the name of the resource group—`Buyers(buy-side)CommandsResourceGroup`. This is the resource group you need to update.

Update the resource group in the role-based policy to include the commands for creating orders

1. Click **Access Management > Resource Groups**.
2. From the list of resource groups, select **Buyers(buy-side)CommandsResourceGroup**.
3. Click **Change** to display the Change Resource Group page.
4. Click **Next** to display the Details page.
5. From the Available Resources list, select the following commands for creating orders:

   ```
   com.ibm.commerce.order.commands.OrderCopyCmd
   com.ibm.commerce.order.commands.OrderScheduleCmd
   com.ibm.commerce.orderitems.commands.OrderItemMoveCmd
   com.ibm.commerce.orderitems.commands.OrderItemUpdateCmd
   com.ibm.commerce.requisitionlist.commands.RequisitionListSubmitCmd
   com.ibm.commerce.orderitems.commands.OrderItemAddCmd
   com.ibm.commerce.orderquotation.commands.OrderQuotationCreateCmd
   ```
6. Click **Add.**
7. Click **Finish**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Orders scenario 2: Allowing only Buyer Administrators to modify orders

Note: This scenario does not apply to WebSphere Commerce Professional Edition.

By default, all users are permitted to modify orders they have created, regardless of their position in their organization. In some cases, you may want only the organization's buyer administrator to have the authority to modify orders.

In this scenario, you will change a resource-level policy, as well as a role-based policy. To allow only buyer administrators to modify orders belonging to members of a buyer organization, you need to do the following:

- Use the Appendix to find the resource-level policy that specifies who can modify an order.
- Change the policy's access group from all users, to those with the `buyer administrator` role.
- Remove the specification of the resource relationship to permit buyer administrators to modify orders belonging to other users.
- Update the policy's name, display name, and description.
- Identify the commands for modifying orders.
- Use the Appendix to find the role-based policy for buyer administrator. This policy defines the commands that users with the buyer administrator role can execute. You must update this policy's resource group to permit buyer administrators to execute the commands for modifying orders.
- Update the role-based policy's resource group to include the commands for modifying orders.

Steps to take

Identify the resource-level policy

1. Look under Orders, in the Appendix, to identify the resource-level policy to be changed. The policy is: `AllUsersExecuteOrderWriteCommandsOnOrderResource`.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. From the list of policies, select **AllUsersExecuteOrderWriteCommandsOnOrderResource**.
5. Note the name of the policy's action group—`OrderWriteCommands`. You need to view this action group to find the name of the command for creating an order.

Change the access group

1. Click **Change** to display the Change Policy page.
2. For User Group, click **Find** and select **Buyer Administrators**.
3. Click **OK**.
4. For Relationship, select **None**.
5. Update the policy's name, display name, and description to reflect the change of access group.
6. Click **OK**.

Identify the commands for modifying orders

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **OrderWriteCommands** .
3. Click **Change** to display the Change Action Group page. Make note of the names of the commands for modifying orders:

   ```
   com.ibm.commerce.order.commands.OrderCancelCmd
   com.ibm.commerce.order.commands.OrderCopyCmd-Write
   com.ibm.commerce.order.commands.OrderUnlockCmd
   com.ibm.commerce.orderitems.commands.OrderItemAddCmd
   com.ibm.commerce.orderitems.commands.OrderItemDeleteCmd
   com.ibm.commerce.orderitems.commands.OrderItemMoveCmd
   com.ibm.commerce.orderitems.commands.OrderItemUpdate.Cmd
   com.ibm.commerce.orderquotation.commands.OrderItemSelectCmd
   ```

   You must add these commands to the resource group that contains the list of commands a buyer can execute.

   Note: When you add the command, `com.ibm.commerce.order.commands.OrderCopyCmd-Write`, to the resource group, it appears under Available Resources as `com.ibm.commerce.order.commands.OrderCopyCmd`.

Identify the role-based policy for the Buyer Administrator role

1. Look under Role-Based Policies in the Appendix to find the role-based policy for buyer administrators. The policy is: `BuyerAdministratorsExecuteBuyersAdministratorsCommands`.
2. Click **Access Management** > **Policies**.
3. For View, select **Root Organization** to display the site-level policies.
4. Locate the policy in the list.
5. Make note of the name of the resource group— `BuyersAdministratorsCommmandsResourceGroup`.

   This is the name of the resource group you need to update.

Update the resource group in the role-based policy to include the commands for modifying orders

1. Click **Access Management > Resource Groups**.
2. Select **BuyersAdministratorsCommandsResourceGroup**.
3. Click **Change** to display the Change Resource Group page.
4. Click **Next** to display the Details page.
5. From the Available Resources list, select the commands for modifying orders:

   ```
   com.ibm.commerce.order.commands.OrderCancelCmd
   com.ibm.commerce.order.commands.OrderCopyCmd
   com.ibm.commerce.order.commands.OrderUnlockCmd
   com.ibm.commerce.orderitems.commands.OrderItemAddCmd
   com.ibm.commerce.orderitems.commands.OrderItemDeleteCmd
   com.ibm.commerce.orderitems.commands.OrderItemMoveCmd
   com.ibm.commerce.orderitems.commands.OrderItemUpdate.Cmd
   com.ibm.commerce.orderquotation.commands.OrderItemSelectCmd
   ```
6. Click **Add** to add the command to the resource group.
7. Click **Finish**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.

4. Click **Update**.

Orders scenario 3: Allowing RMA approvers to approve all RMAs

By default, return merchandise authorization (RMA) approvers for a store are only permitted to approve RMAs for their own stores. In some cases, you may want to allow RMA approvers to approve RMAs for any store. This might be desirable if several stores are owned by the same organization or if the same person handles the RMA approvals for multiple stores.

In this scenario, you will create a new access group and use it in a new resource-level policy. To allow RMA approvers to approve RMAs against any store, you need to do the following:

- Use the Appendix to find the resource-level policy that permits RMA approvers for an organization to approve RMAs for their organization.
- Note the name of the resource group and action group used in the policy.
- View the policy's access group, `RMAApproversForOrg`, and note the roles it includes. The access group is defined using both organizations and roles as selection criteria. To give users authority to perform an action across multiple organizations, the access group must be defined without organizational criteria.
- Create a new access group, `RMAApprovers`, that uses the same roles but does not include the organizational criteria.
- Create a new policy using:
  - The new access group, `RMAApprovers`
  - The action group from the existing policy
  - The resource group from the existing policy

Steps to take

Identify the action group and resource group to use in defining the new policy

1. Look under Orders, in the Appendix, to find the resource-level policy that authorizes `RMAApproversForOrg` to approve RMAs for their stores. The policy is: `RMAApproversForOrgExecuteRMAApproveCommandsOnRMAResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`RMAApproveCommands`. This is the action group you will use in defining your new policy.
6. Note the name of the resource group—`RMADataResourceGroup`, This is the resource group you will use in defining your new policy.
7. Note the name of the access group—`RMAApproversForOrg`. View this access group to see the roles to include in your new access group.

Identify the roles to be used in the new access group

1. Click **Access Management > Access Groups**.
2. From the list of access groups, select **RMAApproversForOrg**.
3. Click **Change**.
4. Select **Criteria** to display the Criteria page.

5. Under Selected Roles and Organizations, note the roles used in the access group:
   - `Customer Service Supervisor`
   - `Seller`
   - `Sales Manager`
   - `Operations Manager`
6. Click **Cancel** to return to the list of access groups.

Define the new access group

1. Click **New** to display the Details page for the new access group.
2. For Name, specify `RMAApprovers`.
3. For Description, specify a description of the access group.
4. For Parent Organization, select Root Organization.
5. Click **Next** to display the Criteria page for the new access group.
6. Click **Criteria based on organizations and roles**.
7. From the list of roles, select the following roles:
   - **Customer Service Supervisor**
   - **Seller**
   - **Sales Manager**
   - **Operations Manager**
8. Click **Finish**.

Define the new policy

1. Click **Access Management > Policies**.
2. Click **New** to display the New Policy page.
3. For Name, specify: `RMAApproversExecuteRMAApproveCommandsOnRMAResource`
4. For Display Name, specify a short description of the policy in your local language.
5. For Description, specify a longer description of what the policy does, in your local language.
6. For User Group, click **Find** and select **RMAApprovers**.
7. Click **OK**.
8. For Resource Group, select **RMADataResourceGroup**.
9. For Action Group, select **RMAApproveCommands**.
10. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Membership scenario 1: Remove the ability of users to self-register

By default users are permitted to self-register if they belong to a registered organization. Membership administrators are also authorized to register users that belong to their organization. For sites that require strictly controlled access, it might be necessary to remove the ability to self-register and require that users be registered by membership administrators.

Note: In WebSphere Commerce Professional Edition, there are only three organizations, Root Organization, Default Organization and Seller Organization.

In this scenario, you will remove the resource-level policy that permits users to self-register but leave in place a policy that permits membership administrators to register users in their organization.

To delete the resource-level policy that allows users to self-register, do the following:

- Use the Appendix to find the resource-level policy that allows users to self-register.
- Delete the policy.

Steps to take

Delete the policy

1. Look under Membership, in the Appendix, to find the resource-level policy that allows users to self-register. The policy is: `GuestsExecuteUserSelfRegistrationCommandsOnOrganizationResource`.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display policies that it owns.
4. From the list of policies, select **GuestsExecuteUserSelfRegistrationCommandsOnOrganizationResource**
5. Click **Delete**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.
5. Repeat steps 3 and 4 for the **Access Control Policy Groups Registry**.

Membership scenario 2: Allowing only registered and approved users to change their address information

By default, users can modify their address information if their registration has been approved or is pending approval. In some cases, you might want only registered and approved users to manage their addresses.

In this scenario, you will change the access group for the resource-level policy that authorizes users to manage their address information, as follows:

- Use the Appendix to find the resource-level policy that allows users to manage their address information.
- Change the access group for the policy.

  Because the access group `RegisteredApprovedUsers` does not contain any roles, you do not need to update a role-based policy for this change.

Steps to take

Change the resource-level policy's access group

1. Look under Membership, in the Appendix, to find the resource-level policy that allows users to manage their address information. The policy is—`NonRejectedUsersExecuteAddressManageCommandsOnUserResource`.

   Note: Non-rejected users are users whose registration has not been rejected. Their registration has either been approved or is pending approval.
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies that it owns.
4. From the list of policies, select **NonRejectedUsersExecuteAddressManageCommandsOnUserResource.**
5. Click **Change** to display the Change Policy page.
6. For User Group, click **Find** and select **RegisteredApprovedUsers**.
7. Click **OK**.
8. Update the policy's name, display name, and description to reflect the change of access group.
9. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Membership scenario 3: Allowing member registrars to register users

By default, membership administrators for an organization are authorized to register member of their organization. The access group, `MemberAdministratorsForOrg`, includes several roles such as buyer administrator and seller administrator, which are authorized to perform a variety of administrative tasks. In some cases, you might want to create a separate role that is authorized only to register organization members:

Here is an overview of the steps involved:

- Create a new role, and for it, a new access group, a new resource group, and a new role-based policy.
- Modify an existing resource-level policy to use the new role.

In this scenario, you will do the following:

- Define a new role called `Member Registrar`.
- Define a new access group, called `MemberRegistrars`, which includes the member registrar role.
- Use the Appendix to find the resource-level policy that permits membership administrators to register members.
- Note the name of the action in its action group. You must create a new resource group with this action and use it in the role-based policy for the new role. Keep

in mind that, in role-based policies for actions, the action group contains only a single action execute. The resource group contains the actions (commands) that can be executed.

- Define a new resource group, called `UserAdminRegistrationCommands`, which includes the command for registering members. You will use this resource group in the role-based policy for the member registrar role.
- Define a new role-based policy for member registrars, which uses the MemberRegistrars access group and the MemberRegistrationCommands resource group.
- Modify the resource-level policy that defines who can register members and change its access group from MembershipAdministrators to MemberRegistrars.

Steps to take

Define the new role

1. From the Organization Administration Console, click **Access Management > Roles**.
2. On the Roles page, click **New**.
3. For Name, specify Member Registrar.
4. For Description, specify a description of the member registrar role in your local language.
5. Click **OK**.

Define a new access group containing the member registrar role

1. Click **Access Management > Access Groups**.
2. On the Access Groups page, click **New** to display the Details page for the new access group.
3. For Name, specify: MemberRegistrars.
4. For Parent Organization, select **Root Organization**.
5. For Description, specify a description of the access group in your local language.
6. Click **Next** to display the Criteria page for the new access group.
7. Click **Based on organizations and roles**.
8. From the Role list, select **Member Registrar**.
9. Click **For Organization** to specify that the role must played within the user's own organization or it's ancestors.
10. Click **Finish**.

Identify the actions to use in the resource group for the member registrar role-based policy

1. Look under Membership, in the Appendix, to find the policy that permits membership administrators to register users. The policy is:
   ```
   CSAMembershipAdministratorsForOrgExecuteUserAdminRegistrationCommandsOn
   OrganizationResource
   ```
2. Click **Access Management > Policies**.
3. For View, select **Root Organization** to display the site-level policies.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`UserAdminRegistration`. This is the action group you need to view to identify the actions for registering members.
6. Click **Access Management > Action Groups**.

7. From the list of action groups, select **UserAdminRegistration**.
8. Click **Change** to display the Change Action Group page.
9. Note the name of the command for registering members: `com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd`.

Define the new resource group to be used in the role-based policy for member registrars

1. Click **Access Management > Resource Groups** to display the Resource Groups page.
2. Click **New** to display the General page for the new resource group.
3. For Name, specify `UserAdminRegistrationCommands`.
4. For Display Name, specify a description of the resource group in your local language.
5. For Description, specify a longer description of the resource group, in your local language.
6. For Type, select **Explicit Resource Group**.
7. Click **Next**.
8. Click **Next** to display the Details page for the new resource group.
9. From the Available Resources list, select the following:
   ```
   com.ibm.commerce.usermanagement.commands.
   UserRegistrationAdminAddCmd
   ```
10. Click **Add**.
11. Click **Finish**.

Define a role-based policy for the member registrar role

1. Click **Access Management** > **Policies**.
2. On the Policies page, click **New**.
3. For Name, specify **MemberRegistrarsExecuteUserAdminRegistrationCommands**.
4. For Display Name, specify a description of the policy in your local language.
5. For Description, specify a longer description of what the policy does, in your local language.
6. For User Group, click **Find** and select **MemberRegistrars**.
7. Click **OK**.
8. For Resource Group, select **UserAdminRegistrationCommands**.
9. For Action Group, select **ExecuteCommandActionGroup**.
10. Click **OK**.

Note: After creating this new policy, it has to be assigned to a policy group before it becomes effective. This must be done through XML. For more information, see Chapter 13, "Customizing access control polices using XML," on page 129.

Modify the resource-level policy to use the new access group

After modifying the resource-level policy, only users who play the Member Registrar role in the same organization as the resource will be allowed to register the user. Users who play the role in any other organization will not be able to do so.

1. From the list of policies, select the following:
   ```
   CSAMembershipAdministratorsForOrgExecuteUserAdmin
   RegistrationCommandsOnOrganizationResource
   ```

.

2. Click **Change** to display the Change Policy page.
3. Update the policy's name, display name and description to reflect the change of access group.
4. For User Group, click **Find** and select **MemberRegistrars**.
5. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Coupons scenario 1: Allowing only buyers to redeem coupons

By default, all users are permitted to redeem coupons. In some cases, you may want to limit coupon redemption to users with the `Buyer` role in WebSphere Commerce.

In this scenario, you will change a resource-level policy, as well as its associated role-based policy. To limit coupon redemption to users with the `Buyer` role, you need to do the following:

- Use the Appendix to find the resource-level policy that specifies who can redeem a coupon.
- Change the policy's access group from all users, to those with the `Buyer` role.
- Identify the command for redeeming coupons.
- Use the Appendix to find the role-based policy for Buyer (buy-side). This policy defines the commands that users with the Buyer (buy-side) role can execute. You must update this policy's resource group to permit buyers to execute the command for redeeming coupons.
- Update this role-based policy's resource group to include the commands for redeeming coupons.

Steps to take

Identify the resource-level policy and its action group

1. Look under Coupons, in the Appendix, to identify the resource-level policy to be changed. The policy is:

   `AllUsersExecuteCouponRedemptionCommandsOnCouponWalletResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display policies it owns.
4. From the list of policies, select the following:

   `AllUsersExecuteCouponRedemptionCommandsOnCoupon`
   `WalletResource`
5. Note the name of the policy's action group— `CouponRedemption`. This is the action group you must view to find the name of the commands for redeeming coupons.

Change the access group

1. Click **Change** to display the Change Policy page.
2. For User Group, click **Find** and select **Buyers (buy-side)**.
3. Click **OK**.
4. Update the policy's name, display name, and description to reflect the change of access group.
5. Click **OK**.

Identify the commands for redeeming coupons

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **CouponRedemption**.
3. Click **Change** to display the Change Action Group page. Note the name of the commands for creating bids:

   ```
   com.ibm.commerce.couponredemption.commands.CouponDSSCmd
    com.ibm.commerce.couponredemption.commands.UseCouponIdCmd
   ```

   You must add these commands to the resource group that contains the list of commands a buyer can execute.

Identify the role-based policy for buyers (buy-side)

1. Look under Role-Based Policies in the Appendix to find the role-based policy for buyers (buy-side). The policy is:

   `Buyers(buy-side)ExecuteBuyers(buy-side)CommandsResourceGroup`
2. Click **Access Management** > **Policies**.
3. For View, select **Root Organization** to display the policies it owns.
4. Locate the policy in the list.
5. Note the name of the resource group: `Buyers(buy-side)CommandsResourceGroup`. This is the name of the resource group you need to update.

Update the resource group in the role-based policy to include the command for creating bids

1. Click **Access Management > Resource Groups**.
2. Select **Buyers(buy-side)CommandsResourceGroup**.
3. Click **Change** to display the Change Resource Group page.
4. Click **Next** to display the Details page.
5. From the Available Resources list, select **com.ibm.commerce.couponredemption.commands.CouponDSSCmd com.ibm.commerce.couponredemption.commands.UseCouponIdCmd**. These are the commands for redeeming coupons.
6. Click **Add** to add the commands to the resource group.
7. Click **Finish**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Coupons scenario 2: Permitting both coupon administrators and Operations Managers to create e-coupon promotions

By default, coupon administrators for a store can create e-coupon promotions for their store. In some cases, you might want to grant this authority to Operations Managers as well.

The flexible design of access control policies offers several methods for implementing this change:

- You can add the Operations Manager role to the access group for the policy that specifies who can create e-coupon promotions.
- You can create a new policy that permits Operations Managers to create e-coupon promotions.

This scenario illustrates the first approach. It shows you how to add the Operations Manager role to the resource-level policy that authorizes coupon administrators to create coupons.

To make this change, you need to do the following:

- Use the Appendix to find the resource-level policy that specifies who can create e-coupon promotions.
- Change the policy's access group to include users with the `Operations Manager` role.
- View the resource-level policy's action group to identify the command for creating e-coupon promotions.
- Use the Appendix to find the role-based policy for an `Operations Manager`. This policy defines the commands that users with the `Operations Manager` role can execute. You must update this policy's resource group to permit store administrators to execute the commands for creating e-coupon promotions.
- Update this role-based policy's resource group to include the command for creating e-coupon promotions.

Steps to take

Identify the action group and access group for the resource-level policy

1. Look under Auctions, in the Appendix, to identify the resource-level policy to be changed. The policy is:

   `CouponAdministratorsForOrgExecuteCouponPromotionCreateCommands OnStoreEntityResource`
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies it owns.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`CouponPromotionCreate`. This is the action group you must view to find the name of the command for creating e-coupon promotions.
6. Note the name of the policy's access group—`CouponAdministratorsForOrg`. This is the access group you must update to include the store administrator role.

Change the access group

1. Click **Access Management > Access Groups**.
2. From the list of access groups, select **CouponAdministratorsForOrg**
3. Click **Change** to display the Details page.
4. Click **Criteria** to display the Criteria page.
5. From the Role list, select **Operations Manager**.
6. Click **For Organization** to specify that the role must be played with the resource's own organization, or its' ancestors.
7. Click **Add**.
8. Click **OK**.

Identify the commands for creating e-coupon promotions

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **CouponPromotionCreate**.
3. Click **Change** to display the Change Action Group page. Note the name of the command for creating e-coupon promotions—`com.ibm.commerce.tools.ecoupon.ECouponPromotionSaveCmd`. You must add this command to the resource group that contains the list of commands an Operations Manager can execute.

Identify the role-based policy for Operations Managers

1. Look under Role-Based Policies in the Appendix to find the role-based policy for Operations Managers. The policy is: `OperationsManagersExecuteOperationsManagersCmdResourceGroup.`
2. Click **Access Management** > **Policies**.
3. For View, select **Root Organization** to display the site-level policies.
4. Locate the policy in the list.
5. Note the name of its resource group—`OperationsManagersCmdResourceGroup`. This is the name of the resource group you need to update.

Update the resource group in the role-based policy to include the command for creating e-coupon promotions

1. Click **Access Management > Resource Groups**.
2. Select **OperationsManagersCmdResourceGroup**.
3. Click **Change** to display the Change Resource Group page.
4. Click **Next** to display the Details page.
5. From the Available Resources list, select `com.ibm.commerce.tools.ecoupon.ECouponPromotionSaveCmd`. This is the command for creating e-coupon promotions.
6. Click **Add**.
7. Click **Finish**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Procurement scenario 1: Allowing procurement shopping cart managers to manage the procurement shopping cart for orders created by their organization

Note: This scenario does not apply to WebSphere Commerce Professional Edition.

By default, procurement shopping cart managers are authorized to manage the procurement shopping cart when they have created the order. In some cases, you might want to extend the authority of procurement shopping cart managers to let them manage the procurement cart for orders created by any member of their organization.

To make this change, you need to do the following:

- Use the Appendix to find the resource-level policy that authorizes procurement shopping cart administrators to manage procurement shopping carts.
- Change the resource relationship for this policy from `creator` to `same organizational entity as creator`.

Steps to take

Change the resource relationship for the resource-level policy

1. Look under Procurement, in the Appendix, to find the resource-level policy that authorizes procurement shopping cart managers to manage procurement shopping carts for orders. The policy is:
   ```
   ProcurementShoppingCartManagersExecuteProcurementShopping
   CartManageOnOrderResource
   ```
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display policies it owns.
4. From the list of policies, select the following:
   `ProcurementShoppingCartManagersExecuteProcurementShopping`
   `CartManageOnOrderResource`
5. Click **Change** to display the Change Policy page.
6. For Relationship, select **sameOrganizationalEntityAsCreator**.
7. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Procurement scenario 2: Allow procurement buyer administrators to submit the procurement shopping cart for orders created by their organization

Note: This scenario does not apply to WebSphere Commerce Professional Edition.

By default, procurement shopping cart managers can save or submit procurement shopping carts if they have created the order. In some cases, you might want to divide the responsibility for these tasks. You could allow procurement shopping cart managers to save procurement shopping carts containing orders they have created but give procurement buyer administrators in the same organization as the order creator the authority to submit the procurement shopping cart. This might be beneficial if you want the procurement buyer administrator to review planned purchases before they are submitted.

To make this change, you need to do the following:
- Use the Appendix to find the resource-level policy that authorizes procurement shopping cart managers to center managers to manage fulfillment centers.
- Remove the action for submitting a procurement shopping cart from the policy's action group.
- Define a new action group containing the command for submitting a procurement shopping cart. You will use this action group to define the new resource-level policy that authorizes procurement buyer administrators to submit procurement shopping carts if they are in the same organization as the creator of the order.
- Create a new resource-level policy that authorizes procurement buyer administrators to submit procurement shopping carts if they are in the same organization as the creator of the order.

Steps to take

Identify the resource-level policy's action group and resource group

1. Look under Procurement, in the Appendix, to find the resource-level policy that authorizes procurement shopping cart managers to manage procurement shopping carts for orders. The policy is:
   ```
   ProcurementShoppingCartManagersExecuteProcurement
   ShoppingCartManageOnOrderResource
   ```
2. From the Organization Administration Console, click **Access Management > Policies**.
3. Locate the policy in the list of policies.
4. Note the name of its action group — `ProcurementShoppingCartManage`. You will update this action group to remove the action for submitting procurement shopping carts.
5. Note the name of its resource group — `OrderDataResourceGroup`. You will use this resource group to define the new resource-level policy .

Update the resource-level policy's action group

1. Click **Access Management > Action Groups**.
2. From the list of action groups, select **ProcurementShoppingCartManage**.
3. Click **Change** to display the Change Action Group page.
4. From the Selected Actions list, select **com.ibm.commerce.me.commands.SubmitShoppingCartCmd**. You will create a new action group with this action and use the action group in your new resource-level policy.
5. Click **Remove**.
6. Click **OK**.

Define a new action group

1. Click **Access Management > Action Groups**.
2. Click **New** to display the New Action Group page.
3. For Name, specify `ProcurementShoppingCartSubmit`.
4. For Display Name, specify a short description of the action group in your local language.
5. For Description, specify a longer description of what the action group does, in your local language.
6. From the Available Actions list, select **com.ibm.commerce.me.commands.SubmitShoppingCartCmd**.
7. Click **Add**.
8. Click **OK**.

Define the new policy

1. Click **Access Management > Policies**.
2. For View, click **Root Organization** to display the policies it owns.
3. Click **New** to display the New Policy page.
4. For Name, specify:
   ```
   ProcurementBuyerAdministratorsExecuteProcurementShoppingCartSubmitCommands
   OnOrderResource
   ```
5. For Display Name, specify a short description of the policy in your local language.
6. For Description, specify a longer description of what the policy does, in your local language.
7. For User Group, click **Find** and select **ProcurementBuyerAdministrators**.
8. Click **OK**.
9. For Resource Group, select **OrderDataResourceGroup**.
10. For Action Group, select **ProcurementShoppingCartSubmit**.
11. For Relationship, select **sameOrganizationalEntityAsCreator**.
12. For Policy Type, select **Groupable Template Policy** to designate the policy as a template policy.
13. Click **OK**.

Update the access control policy registry with your change

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

   Note: After creating the new policy, it has to be assigned to a policy group before it is effective. This is done using XML. See, Chapter 13, "Customizing access control polices using XML," on page 129 for more information.

Inventory scenario 1: Permit fulfillment center managers to update fulfillment centers but not to delete them

By default, fulfillment center managers are authorized to update or delete the fulfillment centers associated with their store. In some cases, you might want to allow fulfillment center managers to update fulfillment centers but not to delete them.

To make this change, you need to do the following:

- Use the Appendix to find the resource-level policy that authorizes fulfillment center managers to manage fulfillment centers.
- Remove the action for deleting a fulfillment center from the policy's action group.

Steps to take

Remove the action for deleting a fulfillment center

1. Look under Procurement, in the Appendix, to find the resource-level policy that authorizes procurement shopping cart managers to manage procurement shopping carts for orders. The policy is:
   ```
   FulfillmentCenterManagersForOrgExecuteFulfillmentCenter
   ManageCommandsOnFulfillmentResource
   ```
2. From the Organization Administration Console, click **Access Management > Policies**.
3. Locate the policy in the list of policies.
4. Note the name of its action group—`FulfillmentCenterManage`. You need to update this action group to remove the action for deleting fulfillment centers.
5. Click **Access Management > Action Groups.**
6. From the list of action groups, select **FulfillmentCenterManage**.
7. Click **Change** to display the Change Action Group page.
8. From the Selected Actions list, select **com.ibm.commerce.inventory.commands.FulfillmentCenterDeleteCmd**.
9. Click **Remove**.
10. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Inventory scenario 2: Permit only logistics managers, operations managers, and account representatives to create, update, or delete fulfillment centers

By default, fulfillment center managers are authorized to create, update or delete the fulfillment centers associated with their store. The fulfillment center manager's access group includes the roles: Seller, Logistics Manager, Operations Manager, and Account Representative. In some cases, you might not want Sellers to be authorized as fulfillment center managers.

To make this change, you need to do the following:

- Use the Appendix to find the resource-level policy that authorizes fulfillment center managers to manage fulfillment centers.
- Remove the seller role from the definition of the `fulfillment center managers` access group.

Steps to take

Remove the seller role from the access group

1. Look under Procurement, in the Appendix, to find the resource-level policy that authorizes procurement shopping cart managers to manage procurement shopping carts for orders. The policy is:

   ```
   FulfillmentCenterManagersForOrgExecuteFulfillmentCenterManage
   CommandsOnFulfillmentResource
   ```
2. From the Organization Administration Console, click **Access Management > Access Groups**.
3. From the list of access groups, select **FulfillmentCenterManagersForOrg**.
4. Click **Change** to display the Change Access Group page.
5. Click **Access Management > Access Groups**.
6. Click **Change** to display the Details page.
7. Click **Criteria** to display the Criteria page.
8. From the Role list, select **Seller**.
9. Click **Remove**.
10. Click **OK**.

Update the access control policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. Click **Update**.

Business intelligence scenario 1: Allowing auditors to view business intelligence reports

By default, intelligence report viewers are permitted to view business intelligence reports for their store. In some cases, you might also want to create a new role called `auditor` and authorize users with this role to view a store's business intelligence reports.

Here is an overview of the steps involved:

- Create a new role, (`Auditor`) and for it, a new access group `Auditors`, a new resource group, and a new role-based policy.
- Add the new role to the resource-level policy's access group.
- Add the `Auditor` role to the access group of the resource-level policy that defines who can view business intelligence reports for their stores.

In this scenario, you will do the following:

- Use the Appendix to find the resource-level policy that permits business intelligence report viewers to view business intelligence reports.

- Note the name of the action in its action group. You must create a new resource group with this action and use it in the role-based policy for the new role. Keep in mind that, in role-based policies for actions, the action group contains only a single action execute. The resource group contains the actions (commands) that can be executed.
- Define a new resource group, called `AuditorCommands`, which includes the command for viewing business intelligence reports. You will use this resource group in the role-based policy for the auditor role.
- Define a new role-based policy for auditors, which uses the Auditors access group and the AuditorCommands resource group.
- Add the auditor role to the access group for the resource-level policy that defines who can view business intelligence reports for their store.

Steps to take

Define the new auditor role

1. From the Organization Administration Console, click **Access Management > Roles**.
2. On the Roles page, click **New**.
3. For Name, specify Auditor.
4. For Description, specify a description of the auditor role in your local language.
5. Click **OK**.

Define a new access group for the auditor role

1. Click **Access Management > Access Groups**.
2. On the Access Groups page, click **New** to display the Details page for the new access group.
3. For Name, specify—Auditors.
4. For Description, specify a description of the access group in your local language.
5. For Parent Organization, select **Root Organization**.
6. Click **Next** to display the Criteria page for the new access group.
7. Click **Based on organizations and roles**.
8. From the Role list, select **Auditor**.
9. Click **Add**.
10. Click **Finish**.

Identify the actions to use in the resource group for the auditor role's role-based policy

1. Look under Business Intelligence, in the Appendix, to find the policy that authorizes intelligence report viewers to view business intelligence reports. The policy is:
   ```
   IntelligenceReportViewersForOrgExecuteViewBusinessIntelligenceReport
   CommandsOnStoreEntityResource
   ```
2. From the Organization Administration Console, click **Access Management > Policies**.
3. For View, select **Root Organization** to display the policies it owns.
4. Locate the policy in the list.
5. Note the name of the policy's action group—`ViewBusinessIntelligenceReport`. This is the action group you must view to identify the actions for registering members.

6. Click **Access Management > Action Groups**.
7. From the list of action groups, select **ViewBusinessIntelligenceReport**.
8. Click **Change** to display the Change Action Group page.
9. Note the name of the command for viewing business intelligence reports—`com.ibm.commerce.bi.commands.BIShowReportCmd`.

Define the new resource group to be used in the role-based policy for the auditor role

1. Click **Access Management > Resource Groups** to display the Resource Groups page.
2. Click **New** to display the General page for the new resource group.
3. For **Name**, specify AuditorCommands.
4. For **Display Name**, specify a description of the resource group in your local language.
5. For **Description**, specify a longer description of the resource group, in your local language.
6. Click **Next**.
7. For Type, select **Explicit Resource Group**.
8. Click **Next** to display the Details page for the new resource group.
9. From the Available Resources list, select **com.ibm.commerce.bi.commands.BIShowReportCmd**.
10. Click **Add**.
11. Click **Finish**.

Define the role-based policy for the auditor role

1. Click **Access Management > Policies**.
2. On the Policies page, click **New**.
3. For Name, specify **AuditorsExecuteAuditorCommands**.
4. For Display Name, specify a description of the policy in your local language.
5. For Description, specify a longer description of what the policy does, in your local language.
6. For User Group, click **Find** and select **Auditors**.
7. Click **OK**.
8. For Resource Group, select **AuditorCommands**.
9. For Action Group, select **ExecuteCommandActionGroup**.
10. Click **OK**.

Add the auditor role to the resource-level policy's access group

1. Click **Access Management > Access Groups.**
2. From the list of access groups, select **IntelligenceReportViewersForOrg**.
3. Click **Change** to display the Change Access Group page.
4. Click **Criteria** to display the Criteria page for the access group.
5. From the Role list, select **Auditor**.
6. Click **For Organization** to specify that the role must be played within the resource's own organization or its' ancestors.
7. Click **Add**.
8. Click **OK**.

Update the policy registry with your changes

1. Log on to the Administration Console.
2. Click **Configuration > Registry**.
3. From the list of registries, select **Access Control Policies**.
4. **Click Update.**

Chapter 13. Customizing access control polices using XML

The WebSphere Commerce Administration Console allows you to make simple changes to access control policies and their parts. To make more sophisticated changes, you need to edit the XML files directly, and then load them into the database.

Before you begin making changes to the XML files for access control, you should read the chapter on access control in *WebSphere Commerce Programming Guide and Tutorials*. This chapter provides a technical overview of access control and explains how to create customized commands, entity beans, and JSP templates that can be protected by access control policies.

Once you have finished the code customizations following the guidance provided in the *WebSphere Commerce Programming Guide and Tutorials*, you can edit the XML files for access control to establish the protections you require.

Changes that can only be made by editing and loading the XML files

The following changes can only be made by editing and then loading the appropriate XML files:

- Creating or modifying an action
- Creating or modifying a relationship
- Creating or modifying a relationship group
- Creating or modifying a resource
- Creating or modifying attributes
- Creating or modifying access groups using complex criteria
- Creating or modifying resource groups using complex criteria
- Creating a role-based policy for views
- Changing the action group in a role-based policy for views
- Creating or modifying a policy group
- Associating policies with policy groups

About the XML files for access control

The names and descriptions of WebSphere Commerce's XML files, DTD files, and XSL files for the XML Transformer, are shown in the following table.

Table 12. WebSphere Commerce XML files for access control

File Name	Description
ACUserGroups_de_DE.xml ACUserGroups_en_US.xml ACUserGroups_es_ES.xml ACUserGroups_fr_FR.xml ACUserGroups_it_IT.xml ACUserGroups_ja_JP.xml ACUserGroups_ko_KR.xml ACUserGroups_pt_BR.xml ACUserGroups_zh_CN.xml ACUserGroups_zh_TW.xml	Access group definitions and descriptions in each supported language.
defaultAccessControlPolicies.xml	Main file containing the definitions of default access control policies, action groups, resource groups, relationships, relationship groups, actions, resource categories, and attributes.
defaultAccessControlPolicies_de_DE.xml defaultAccessControlPolicies_en_US.xml defaultAccessControlPolicies_es_ES.xml defaultAccessControlPolicies_fr_FR.xml defaultAccessControlPolicies_it_IT.xml defaultAccessControlPolicies_ja_JP.xml defaultAccessControlPolicies_ko_KR.xml defaultAccessControlPolicies_pt_BR.xml defaultAccessControlPolicies_zh_CN.xml defaultAccessControlPolicies_zh_TW.xml	Files containing the display names and descriptions for default access control policies, action groups, actions, resource groups, resource categories, relationships, and attributes, in each supported language.
ACPoliciesfilter.xml	Filter file used in the extraction of all access control information from the database.
OrganizationPoliciesFilter.xml	Filter file used in the extraction of all the access control information related to policies owned by a specific organization.
ACUserGroupsFilter.xml	Filter file used in the extraction of all the access group information.
accesscontrolpolicies.dtd	The access control policies XML file must conform to this DTD.

Table 12. WebSphere Commerce XML files for access control (continued)

File Name	Description
`accesscontrolpoliciesnls.dtd`	The access control policies NLS (national language specific) XML file (display names and descriptions only) must conform to this DTD.
`ACUserGroups_en_US.dtd`	The access control user groups XML file must conform to this DTD.
`accesscontrol.xsl`	The XSL transform rule file for the access control policies XML file.
`accesscontrolnls.xsl`	The XSL transform rule file for the access control policies NLS XML file (display names and descriptions only).
`ACUserGroup.xsl`	The XSL transform rule file for the access group XML files.
`wcstoacpolicies.xsl`	The XSL transform rule file for the `ExtractedACPolicies.xml` file after extract, to create the access control policies XML file.
`wcstoacpoliciesnls.xsl`	The XSL transform rule file for the `ExtractedACPolicies.xml` after extract, to create the access control policies NLS XML file.
`wcstoacusergroup.xsl`	XSL transform rule file for the `ExtractedACPolicies.xml` file after extract, to create the access group XML file

Changing the XML files

You can manipulate the XML files to perform the following authorization tasks:

- Protecting views
- Protecting controller commands
- Implementing resource-level access control
- Protecting databeans
- Grouping resources by attributes
- Defining relationships
- Defining relationship groups

Protecting views

Any view that is called directly from an URL, or that is launched as a redirect from another command, needs a role-based access control policy in order to be displayed. The following example displays a role-based policy for views:

```
<Policy Name="ProductManagersExecuteProductManagersViews"
OwnerID="RootOrganization"
UserGroup="ProductMangers"
ActionGroupName="ProductMangersViews"
ResourceGroupName="ViewCommandResourceGroup"
PolicyType="groupableStandard">
</Policy>
```

The ResourceGroup name, ViewCommandResourceGroup, indicates that this is a role-based policy for views. The policy states that users in the ProductManagers user group, can display the views in the ProductMangersViews action group. Similarly, for most roles, there is a corresponding action group which groups the views that that role can access, such as Seller role -> Sellers access group -> SellersViews action group.

The following is an example of the ProductMangersViews action group:

```
<ActionGroup Name="ProductManagersViews"
 OwnerID="RootOrganization">

<ActionGroupAction Name="ProductImageView"/>
<ActionGroupAction Name="ProductManufacturerView"/>
<ActionGroupAction Name="ProductSalesTaxView"/>

 </ActionGoup>
```

The example above lists the three actions, ProductImageView, ProductManufacturerView, and, ProductSalesTaxView that can be performed in the ProductManagerViews action group.

The following is an example of the ProductImageView action definition:

```
<Action Name="ProductImageView"
CommandName="ProductImageView">
</Action>
```

The Name attribute, ProductImageView,is used as a tag for referencing the action elsewhere in the XML such as when associating the action with an action group.

Note: The name of the view, stored in the VIEWNAME column of the VIEWREG table, must match the CommandName in the action definition. The value of CommandName is stored in the ACTION column of the ACACTION table. The Name and CommandName attributes do not have to be the same.

Adding a new view using existing policies

To add a new view that is accessible by roles with existing role-based View policies, create an XML file similar to the one shown and then do the following:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">

<Policies>

 <Action Name="MyNewView"
         CommandName="MyNewView">
  </Action>

  <ActionGroup Name="ProductManagersViews" OwnerID="RootOrganization">
             <ActionGroupAction Name="MyNewView"/>
   </ActionGroup>

</Policies>
```

1. Create a new action definition in the XML file that has the view name *MyNewView*. This can be any name that you choose.

   ```
   <Action Name="MyNewView"
      CommandName="MyNewView">
   </Action>
   ```

2. Determine which roles should have access to this view, and associate the new action with the corresponding action groups in the XML file as in the following example:

```
<ActionGroup Name="ProductManagersViews"
    OwnerID="RootOrganization">

  <ActionGroupAction Name="MyNewView"/>

 </ActionGroup>
```

There is already a role-based policy, `ProductManagersExecuteProductManagersViews`, that includes this action group, so a new policy does not have to be created. Also, the default role-based policies belong to the `ManagementAndAdministrationPolicyGroup` policy group which applies to most, if not all, of the organizations in the site, so no further policy group subscription is necessary.

3. Load your XML changes into the database. For more information on loading the XML changes, see "Loading your changes into the database" on page 161.
4. Update the Access Control Policies Registry in the Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.
   c. From the list of registries, select **Access Control Policies**.
   d. Click **Update**.

Adding a new view using a new policy

To add a new view that is accessible by a new role that does not have an existing role-based policy, create an XML file similar to the one shown, and then do the following:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">

<Policies>

 <Action Name="MyNewView"
         CommandName="MyNewView">
  </Action>

  <ActionGroup Name="XYZViews" OwnerID="RootOrganization">
             <ActionGroupAction Name="MyNewView"/>
   </ActionGroup>

 <Policy Name="XYZExecuteXYZViews"
   OwnerID="RootOrganization"
   UserGroup="XYZ"
   ActionGroupName="XYZViews"
   ResourceGroupName="ViewCommandResourceGroup"
   PolicyType="groupableStandard">
 </Policy>

 <PolicyGroup Name="ManagementAndAdministrationPolicyGroup" OwnerID="RootOrganization">
  <PolicyGroupPolicy Name="XYZExecuteXYZViews" PolicyOwnerId="RootOrganization" />
 </PolicyGroup>

</Policies>
```

1. Create a new action definition in the XML file that has the view name *MyNewView*. This can be any name that you choose.
   ```
   <Action Name="MyNewView
   CommandName="MyNewView">
   </Action>
   ```
2. Create a new action group to be associated with the new role:
   ```
   <ActionGroupName="XYZViews"
       OwnerID="RootOrganization">
   </ActionGroup>
   ```

Where *XYZViews* is the name of your action group. The OwnerID for action groups should always be RootOrganization.

3. Associate the new action with the new action group:

```
< ActionGroupName="XYZViews"
     OwnerID="RootOrganization">

 <ActionGroupAction Name="MyNewView"/>

    </ActionGroup>
```

Where *XYZViews* is your action group, and *MyNewView* is the action you created.

4. Create a policy that references the new action group:

```
<Policy Name="XYZExecuteXYZViews"
OwnerID="RootOrganization"
UserGroup="XYZ"
ActionGroupName="XYZViews"
ResourceGroupName="ViewCommandResourceGroup"
PolicyType="groupableStandard">
</Policy>
```

Where *XYZExecuteXYZViews* is your policy name and *XYZViews* is your action group. In WebSphere Commerce 5.5, because of the policy subscription model, the OwnerID for groupable standard and groupable template policies is not used to determine to which resources a policy will apply. The OwnerID value is currently used only by the Administration Console when viewing policies by organization (owner). If a policy is to apply to multiple organizations, it is recommended that the OwnerID be set to the common ancestor organization such as Root Organization. If a policy is to apply only to a specific organization, it is recommended that the OwnerID be set to that organization's `orgentity_id`.

5. Include the new policy in the appropriate policy group. By default, most role-based policies are put into ManagementAndAdministrationPolicyGroup, which should be applied to all organizations.

```
<PolicyGroupName="ManagementAndAdministrationPolicyGroup"
OwnerID="RootOrganization">
<PolicyGroupPolicy Name="XYZExecuteXYZViews" PolicyOwnerId="RootOrganization"/>
</PolicyGroup>
```

Where the `PolicyOwnerId` value must be the same as the `OwnerID` value used in the policy definition.

6. Load your XML changes into the database. For more information on loading the XML changes, see "Loading your changes into the database" on page 161.
7. Update the Access Control Policies Registry in the Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.
   c. From the list of registries, select **Access Control Policies**.
   d. Click **Update**.

You can now use your view.

Protecting controller commands

All controller commands require a role-based access control policy in order to be executed. A controller or task command also requires a resource-level policy if the command is doing resource-level checking. For more information see “Protecting resources” on page 141. The following example displays a role-based policy for controller commands:

```
<Policy Name="SellersExecuteSellersCmdResourceGroup"
   OwnerID="RootOrganization"
   UserGroup="Sellers"
   ActionGroupName="ExecuteCommandActionGroup"
   ResourceGroupName="SellersCmdResourceGroup"
PolicyType="groupableStandard">
</Policy>
```

The ActionGroupName, `ExecuteCommandActionGroup`, indicates that this is a role-based policy for controller commands. The policy states that users in the `Sellers` access group can execute the commands in the `SellersCmdResourceGroup`, resource group.

The following is an example of the `SellersCmdResourceGroup` resource group definition:

-
  ```
  <ResourceGroup Name="SellersCmdResourceGroup" OwnerID="RootOrganization">
  <ResourceGroupResource Name="com.ibm.contract.commands.Contract
  CancelCmdResourceCategory"/>
  <ResourceGroupResource Name="com.ibm.contract.commands.Contract
  CloseCmdResourceCategory"/>
  <ResourceGroupResource Name="com.ibm.contract.commands.Contract
  CreateCmdResourceCategory"/>
  </ResourceGroup>
  ```

The example above shows the following three resources in the resource group, that each correspond to a controller command:

- `com.ibm.contract.commands.ContractCancelCmdResourceCategory`
- `com.ibm.contract.commands.ContractCloseCmdResourceCategory`
- `com.ibm.contract.commands.ContractCreateCmdResourceCategory`

The following is a sample definition of a resource:

```
<ResourceCategory Name="com.ibm.commerce.contract.commands.Contract
CloseCmdResourceCategory"
ResourceBeanClass="com.ibm.commerce.contract.commands.ContractCloseCmd">

<ResourceAction Name="ExecuteCommand"/>

</ResourceCategory>
```

The `Name` attribute, `com.ibm.commerce.contract.commands.ContractCloseCmdResourceCategory`, is used as a tag to refer to the resource in the XML file. The ResourceAction Name, `ExecuteCommand`, is used to specify the actions that can operate on the resource. This information is used in the Administration console when using access control policies to populate the Action selection box that corresponds to a particular resource. In this case, the action `Execute` is specified. The `Execute` action is defined in the following:

```
<Action Name="ExcecuteCommand
CommandName="Execute">
</Action>
```

Note: The interface name of the controller command must match the `ResourceBeanClass` in the resource definition. The value of the `ResourceBeanClass` is stored in the `RESCLASSNAME` column of the `ACRESCGRY` table. These commands can be used as resources because they extend the

ControllerCommand interface, which extends the AccCommand interface, which in turn extends the Protectable interface. For more information on these interfaces refer to the *WebSphere Commerce Programming Guide and Tutorials*.

Adding a new controller command using existing policies

To add a new controller command to be accessed by a new role, that has an existing role-based policy, create an XML file, similar to the one shown. The specific steps are listed afterwards.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">

<Policies>

  < ResourceCategory  Name="com.xyz.commands.MyNewControllerCmdResourceCategory"
    ResourceBeanClass="com.xyz.commands.MyNewControllerCmd">

     <ResourceAction Name="ExecuteCommand"/>
  </ResourceCategory>

        <ResourceGroup Name="SellersCmdResourceGroup"  OwnerID="RootOrganization">
    ResourceGroupResource Name="com.xyz.commands.MyNewControllerCmdResource
    Category"/>
    </ResourceGroup>

</Policies>
```

1. Create a new resource definition in the XML file that corresponds to the interface name of the controller command.

   ```
   <ResourceCategory  Name="com.xyz.commands.MyNewControllerCmdResourceCategory"
      ResourceBeanClass="com.xyz.commands.MyNewControllerCmd">

     <ResourceAction Name="ExecuteCommand"/>
    </ResourceCategory>
   ```
2. Determine which roles should have access to the command and associate the new resource with the corresponding resource groups in the XML file, as in the following example:

   ```
    <ResourceGroup Name="SellersCmdResourceGroup"  OwnerID="RootOrganization">
       <ResourceGroupResource Name="com.xyz.commands.
   MyNewControllerCmdResourceCategory"/>

    </ResourceGroup>
   ```

   You can change the resource group depending on which role you want to use. For more information on role-based policies see, "Role-Based Policies" on page 202.
3. Load your XML changes into the database. For more information on loading the XML changes, see "Loading your changes into the database" on page 161.
4. Update the Access Control Policies Registry in the Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.
   c. From the list of registries, select **Access Control Policies**.
   d. Click **Update**.

Since there is already a role-based policy that includes this resource group, you can now use your new controller command, if it is not doing any resource-level checking. For information on resource-level checking and commands, see "Modifying the resource-level access control of an existing policy" on page 139.

Adding a new controller command using a new policy

To add a new controller command to be accessed by a new role, that does not have an existing role-based policy, create an XML file, similar to the one shown. The specific steps are listed afterwards.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">
<Policies>

  < ResourceCategory  Name="com.xyz.commands.MyNewControllerCmdResourceCategory"
    <ResourceBeanClass="com.xyz.commands.MyNewControllerCmd">

     <ResourceAction Name="ExecuteCommand"/>
  </ResourceCategory>

        <ResourceGroup Name="XYZCmdResourceGroup"  OwnerID="RootOrganization"
    <ResourceGroupResource Name="com.xyz.commands.MyNewController
    CmdResourceCategory"/>
    </ResourceGroup>

    <Policy Name="XYZExecuteXYZsCmdResourceGroup"
    OwnerID="RootOrganization"
    UserGroup="XYZ"
    ActionGroupName="ExecuteCommandActionGroup"
    ResourceGroupName="XYZCmdResourceGroup"
    PolicyType="groupableStandard">
  </Policy>

  <PolicyGroup Name="ManagementAndAdministrationPolicyGroup"
   OwnerID="RootOrganization">
  <PolicyGroupPolicy  Name="XYZExecuteXYZsCmdResourceGroup"
   PolicyOwnerId="RootOrganization" />
 </PolicyGroup>

</Policies>
```

1. Create a new resource definition in the XML file that corresponds with the interface name of the controller command. See "Adding a new controller command using existing policies" on page 136 step one, for an example.
2. Create a new resource group to be associated with the new role:
   ```
   <ResourceGroup Name="XYZCmdResourceGroup" OwnerID="RootOrganization">
   </ResourceGroup>
   ```
3. Associate the new resource with the new resource group:
   ```
   <ResourceGroup Name="XYZCmdResourceGroup" OwnerID="RootOrganization">
   <ResourceGroupResource Name="com.xyz.commands.MyNewControllerResourceCategory"/>
   </ResourceGroup>
   ```
4. Create a policy that references your new resource group:
   ```
   <Policy Name="XYZExecuteXYZsCmdResourceGroup"
      OwnerID="RootOrganization"
      UserGroup="XYZ"
      ActionGroupName="ExecuteCommandActionGroup"
      ResourceGroupName="XYZCmdResourceGroup">
      PolicyType="groupableStandard">
   </Policy>
   ```
5. Load your XML changes into the database. For more information on loading the XML changes, see "Loading your changes into the database" on page 161.
6. Update the Access Control Policies Registry in the Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.

c. From the list of registries, select **Access Control Policies**.
d. Click **Update**.

You can now use your controller command if it is not doing any resource-level checking. For information on resource-level checking and commands, see "Modifying the resource-level access control of an existing policy" on page 139.

Modifying the command-level access control for controller command

Based on the default access control policies, the `UserRegistrationAdminAddCmd` command cannot be run by users who only have the Marketing Manager role. The following scenario describes the steps needed to modify the existing policies so that these users can perform this command. You can use the steps in this scenario and customize them to your own requirements.

All controller commands require a command-level access control policy, which has the `ActionGroupName = ExecuteCommandActionGroup`. It also must have a resource group that includes the interface name of the controller command. These policies usually refer to a specific role, for example, `MarketingManagersExecuteMarketingManagerCmdResourceGroup`.

```
<Policy Name="MarketingManagersExecuteMarketingManagerCmdResourceGroup"
   OwnerID="RootOrganization"
   UserGroup="MarketingManagers"
   ActionGroupName="ExecuteCommandActionGroup"
   ResourceGroupName="MarketingManagerCmdResourceGroup"
   PolicyType="groupableStandard">

</Policy>
```

Note: The above policy is one of the default policies that is loaded into the database during instance creation. For more information on the default policies, see "Default access control policies and groups," on page 201.

In this case, if you want users with the Marketing Manager role to be able to execute the `UserRegistrationAdminAddCmd`, you have to add this command to the existing Resource Group used in the policy by creating your own XML file, and do the following:

1. Redefine the `ExecuteCommand` action
2. Redefine `com.ibm.commerce.usermanagement.commands.UserRegistrationAddCmd` as a resource category.
3. Associate the resource category with the required resource group, in this case `MarketingManagerCmdResourceGroup`.
4. Copy the XML file to `*WC_installdir*/xml/policies/xml`. The following is an example of what your XML could look like:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>


<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">
<Policies>

 <Action Name="ExecuteCommand"
   CommandName="Execute">
  </Action>

 <ResourceCategory
      Name="com.ibm.commerce.usermanagement.commands.UserRegistrationAdmin
AddCmdResourceCategory"
   ResourceBeanClass="com.ibm.commerce.usermanagement.commands.
UserRegistrationAdminAddCmd">
  <ResourceAction Name="ExecuteCommand"/>
 </ResourceCategory>

 <ResourceGroup Name="MarketingManagerCmdResourceGroup"
```

```
     OwnerID="RootOrganization"
     ResourceGroupResource
         Name="com.ibm.commerce.usermanagement.commands.
   UserRegistrationAdminAddCmdResourceCategory"/>
    </ResourceGroup>

   </Policies>
```

5. Load the XML file into the database using the *WC_installdir*/bin/acpload script. For more information on loading your XML files, see "Loading your changes into the database" on page 161.
6. Update the Access Control Policy Registry in the WebSphere Commerce Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.
   c. From the list of registries, select **Access Control Policies**.
   d. Click **Update**.

You can now use your controller command if it is not doing any resource-level checking. If it is doing resource-level checking, see "Modifying the resource-level access control of an existing policy."

Modifying the resource-level access control of an existing policy: For commands that require resource level access control, they return the protected resource(s) that they are going to access in the command's getResources() method. This triggers a resource level access control check by the WebSphere Commerce access control framework. WebSphere Commerce will search for an access control policy in the system with an Action Group that includes the action that is equal to the current command; in this example com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd. The policy's Resource Group must also include the resource that was returned in the getResources() method. In this case, the UserRegistrationAdminAddCmd command does implement the getResources() method and it returns the organization to which the new user is going to be registered.

Out of the box, in defaultAccessControlPolicies.xml, com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd is already defined as an action:

```
<Action Name="com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd"
   CommandName="com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd">
 </Action>
```

It is also included in an action group, defined in the defaultAccessControlPolicies.xml XML file:

```
<ActionGroup Name="UserAdminRegistration"
    OwnerID="RootOrganization">

  <ActionGroupAction
         Name="com.ibm.commerce.usermanagement.commands.UserRegistrationAdminAddCmd"/>
 </ActionGroup>
```

This action group is already used in an existing bootstrap policy:

```
<Policy
   Name="MembershipAdministratorsForOrgExecuteUserAdminRegistrationCommandsOnOrganizationResource"
   OwnerID="RootOrganization"
   UserGroup="MembershipAdministratorsForOrg"
   ActionGroupName="UserAdminRegistration"
   ResourceGroupName="OrganizationDataResourceGroup"
   PolicyType="groupableTemplate">

    </Policy>
```

Note: Many polices are default polices and loaded into the database during instance creation. For more information on the default policies, see "Default access control policies and groups," on page 201.

To add the required role to the `UserRegistrationAdminAddCmd` do the following:

1. Add the required role to the access group used by the policy. In this example, `MembershipAdministratorsForOrg`.

   This access group is defined in *WC_installdir*/`xml/policies/xml/ACUserGroup_en_US.xml` as follows:

   ```
   <UserGroup Name="MembershipAdministratorsForOrg" OwnerID="RootOrganization"
      Description="Administrators of membership for the organization" MemberGroupID="-97"

   <UserCondition><![ CDATA [
       <profile>
       <orListCondition>
         <simpleCondition>
           <variable name="role"/>
           <operator name="="/>
           <value data="Buyer Administrator"/>
        <qualifier name="org" data="?"/>
         </simpleCondition>
         <simpleCondition>
           <variable name="role"/>
           <operator name="="/>
          <value data="Seller Administrator"/>
        <qualifier name="org" data="?"/>
         </simpleCondition>
      </orListCondition>
       </profile>
    ]]></UserCondition>
   </UserGroup>
   ```

   In the above XML, users are included that have at least one of the specified roles, Buyer Administrator or Seller Administrator for an organization that is an ancestor of the owner of the resource (organization) returned by `getResources()`. If you wanted to add the Marketing Manager role, you would have to enhance it to also include the new role.

2. Copy the XML file to *WC_installdir*/`xml/policies/xml`. The following is an example of what your XML could look like:

   ```
   ?xml version="1.0" encoding="UTF-8"?
   <!DOCTYPE UserGroups SYSTEM "../dtd/ACUserGroups_en_US.dtd">

   <UserGroups>

   <UserGroup Name="MembershipAdministratorsForOrg" OwnerID="RootOrganization"
       Description="Administrators of membership for the organization" MemberGroupID="-97">

     <UserCondition><![CDATA[
        <profile>
       <orListConditio>
         <simpleCondition>
           <variable name="role"/>
           <operator name="="/>
           <value data="Buyer Administrator"/>
        <qualifier name="org" data="?"/>
         </simpleCondition>
         <simpleCondition>
           <variable name="role"/>
           <operator name="="/>
           <value data="Seller Administrator"/>
        <qualifier name="org" data="?"/>
         </simpleCondition>
         <simpleCondition>
           <variable name="role"/>
           <operator name="="/>
           <value data="Marketing Manager"/>
        <qualifier name="org" data="?"/>
         </simpleCondition>
      </orListCondition>
       </profile>
    ]]></UserCondition>
   </UserGroup>

   </UserGroups>
   ```

3. Load the XML file into the database using the `WC_installdir/bin/acpload` script. For more information on loading your XML files, see "Loading your changes into the database" on page 161.
4. Update the Access Control Policy Registry in the WebSphere Commerce Administration Console by doing the following:
   a. Logon to the Administration Console as a Site Administrator.
   b. Click **Configuration** > **Registry**.
   c. From the list of registries, select **Access Control Policies**.
   d. Click **Update**.

Protecting resources

You can add resource-level access control to controller or task commands. Resource-level checking is done at WebSphere Commerce runtime, based on data returned by the `getResources()` method of a command. Resource-level checking can also be done during the `performExcecute()` portion of the command by making direct calls to the access control policy manager using the method `void checkIsAllowed(Object resource, String action) throws ECException`. This method will throw the `ECApplicationException` if the current user is not allowed to perform the specified action on the specified resource.

Note: By default, the `getResources()` method returns null, and no resource—level checking is done.

You need to create a resource-level policy for new commands in the following instances:
- The new command extends from a base WebSphere Commerce command that is doing a resource-level check, and has a resource-level policy, and the new command is implementing a different interface than the base command.
- The new command itself does resource-level access control checking.

The following is an example of a resource-level policy:

```
<Policy Name="ContractMangersForOrgExecuteContractManageCommandsOnContractResource"
  OwnerID="RootOrganization"
  UserGroup="ContractManagersForOrg"
  ActionGroupName="ContractManage"
  ResourceGroupName="ContractDataResourceGroup"
  PolicyType="groupableTemplate">
</Policy>
```

Where:

`Name`: The name of the policy.

`PolicyType`: The policy type. This is a groupable template policy and will dynamically apply to the organizational entity that owns the resource and it's ancestors.

`OwnerID`: The member that owns the policy.

`UserGroup`: The policy applies to users of this group. The naming convention for access groups where roles are dynamically scoped to the organization that owns the resource, is to append `ForOrg` to the group name

`ActionGroupName`: The name of the action group that contains the actions to be performed on the resource.

ResourceGroupName: The name of the resource group that contains the resources to be acted upon.

In the example above, the action group ContractManage is an action group that contains the set of commands that will act on the ContractDataResourceGroup. The following is an example of the action group that is used in the above resource-level policy:

```
<ActionGroupName="ContractManage" OwnerID="RootOrganization">
<ActionGroupActionName="com.ibm.commerce.contract.commands.ContractCancelCmd"/>
<ActionGroupActionName="com.ibm.commerce.contract.commands.ContractCloseCmd"/>
<ActionGroupActionName="com.ibm.commerce.contract.commands.ContractDeleteCmd"/>
</ActionGroup>
```

The commands that were previously defined as resources for role-based policies are now defined as actions. The following is a sample definition of an action that is a part of the above ContractManage group:

```
<Action Name="com.ibm.commerce.contract.commands.ContractCloseCmd"
CommandName="com.ibm.commerce.contract.commands.ContractCloseCmd">
</Action>
```

Note: The value of CommandName should correspond to the interface name of the command that is doing the resource-level check.

Most commands work with enterprise beans. These beans are usually the resources that the resource-level policies are protecting. The following is a sample definition of the resource group that is used in the above resource policy:

```
<ResourceGroup Name="ContractDataResourceGroup" OwnerId="RootOrganization">
<ResourceGroupResource Name="com.ibm.commerce.contract.
objects.ContractResourceCategory"/>
</ResourceGroup>
```

In this example, ContractDataResourceGroup is defined and is composed of one resource. The resource is defined as follows:

```
<ResourceCategory Name="com.ibm.commerce.contract.objects.ContractResourceCategory"
ResourceBeanClass="com.ibm.commerce.contract.objects.Contract"
 <ResourceAction Name="com.ibm.commerce.contract.commands.ContractCancelCmd"/>
 <ResourceAction Name="com.ibm.commerce.contract.commands.ContractCloseCmd"/>
 <ResourceAction Name="com.ibm.commerce.contract.commands.ContractDeleteCmd"/>
 </ResourceCategory>
```

Where:

Name: A tag used to reference this resource elsewhere in the XML file.

ResourceBeanClass: The class representing the resource to protect. This class must implement the Protectable interface. If the resource is an enterprise bean, it's remote interface should extend the Protectable interface.

ResourceAction: Specifies the actions that will be operating on this resource. This information is used by the Administration Console when determining which actions are valid with a particular resource.

Note: For more information on the Protectable interface, see the *WebSphere Commerce Programming Guide and Tutorials*.

Protecting data beans

Data beans contain information about business objects and are used to display object information on a web page. Dynamic web pages are usually mapped to views within WebSphere Commerce, and these views are protected by role-based policies. It is sometimes necessary to further protect the content of the web page by protecting it’s data beans, if they exist.

When data beans are populated using the `DataBeanManager.activate(..)` method, the data bean managers enforce access control on them. Data beans can be protected directly or indirectly, using the Delegator interface. Directly protected data beans also implement the Protectable interface. If an indirectly protected data bean does not implement the Delegator interface, or returns a null value for the `getDelegate() method`, it is not protected and can be displayed by anyone.

Note: For more information on the Protectable interface, refer to the *WebSphere Commerce Programming Guide and Tutorials*.

The following is an example of a resource-level policy for a data bean:

```
<Policy Name="AllUsersDisplayOrderDataBeanResourceGroup"
   OwnerID="RootOrganization"
   UserGroup="AllUsers"
   ActionGroupName="DisplayDataBeanActionGroup"
   ResourceGroupName="OrderDataBeanResourceGroup"
   RelationName="creator"
   PolicyType="groupableStandard">
  </Policy>
```

The ActionGroupName, `DisplayDataBeanActionGroup`, indicates that this policy is a policy for data beans. This action group includes one `Display` action.

Where:

`Name`: The name of this policy.

`UserGroup`: The access group that contains the users to whom the policy applies. In this case, it includes all users.

`ActionGroupName`: The value `DisplayDataBeanActionGroup` indicates that is is a resource-level policy for data beans.

`ResourceGroupName`: The name of the resource group that contains the data beans to be protected.

`RelationName`: The relationship that must be fulfilled between a user and the resource. In this case, the user must be the creator of the business `Order` resource.

The `OrderDataBeanResourceGroup` is defined as follows:

```
<ResourceGroup Name="OrderDataBeanResourceGroup" OwnerID="RootOrganization">
<ResourceGroupResource Name="com.ibm.commerce.order.beans.
OrderListDataBeanResourceCategory"/>
<ResourceGroupResource Name="com.ibm.commerce.order.beans
.OrderDataBeanResourceCategory"/>
</ResourceGroup>
```

The `OrderDataBeanResourceGroup` consists of two resources. The following is a sample resource definition for a DataBean:

```
<ResourceCategory Name="com.ibm.commerce.order.beans.OrderDataBeanResourceCategory"
ResourceBeanClass="com.ibm.commerce.order.beans.OrderDataBean">
<ResourceAction Name="DisplayDataBean"/>
</ResourceCategory>
```

Where:

`Name`: A tag used to refer to this resource in the XML file.

`ResourceBeanClass`: The class name of the databean that is being directly protected. This class must implement the Protectable interface.

`ResourceAction`: An element needed for policy editing in the Administration Console. In this case, this element indicates that `Display` is the valid action to be performed on this resource.

Grouping resources by attributes

Resource groups can be defined entirely by using the `CONDITIONS` column in the `ACRESGRP` table. The CONDITIONS column stores the XML document containing the constraints and attribute value pairs used for grouping resources. This type of resource group is called an implicit resource group, and is usually used when the class name of the resource is not sufficient. For example, if an access control policy applies to `Order` resources that have a status equal to `P` (pending) or `E` (editing by a customer service representative), a resource group can be defined for this.

Note: In order to group resources by attributes other than class name, the resource must implement the Groupable interface. For more information on the Groupable interface, refer to the *WebSphere Commerce Programming Guide and Tutorials*.

The following is an example of the `Order` resource group:

```
<ResourceGroup  Name="OrderResourceGroupwithPEStatus"
      OwnerID="RootOrganization">
  <ResourceCondition>
   <![CDATA[
    <profile>
     <andListCondition>
   <orListCondition>
        <simpleCondition>
         <variable name="Status"/>
         <operator name="="/>
         <value data="P"/>
        </simpleCondition>
        <simpleCondition>
         <variable name="Status"/>
         <operator name="="/>
         <value data="E"/>
        </simpleCondition>
    </orListCondition>
        <simpleCondition>
         <variable name="classname"/>
         <operator name="="/>
         <value data="com.ibm.commerce.order.objects.Order"/>
        </simpleCondition>
      </andListCondition>
     </profile>
   ]]>
  </ResourceCondition>

 </ResourceGroup>
```

Where:

Name: The name of the resource group stored in the GRPNAME column of the ACRESGRP table.

OwnerID: The owner of the resource group. This must be the root organization.

<ResourceCondition>: Specifies the data that will be loaded to the CONDITIONS column of the ACRESGRP table, to define the resource group.

<![CDATA[...]]>: Signifies a section of character data that are used exactly as they are typed .

<profile>: A required parameter for all resource conditions.

An essential component of the resource group definition is the <simpleCondition> element that has name="classname". This element identifies the java class of the resource that the group applies to. The java class, com.ibm.commerce.order.objects.Order, can be seen in the following example:

```
<simpleCondition>
   <variable name="classname"/>
   <operator name="="/>
   <value data="com.ibm.commerce.order.objects.Order"/>
   </simpleCondition>
```

The following example specifies the condition on the com.ibm.commerce.objects.order.objects.Order resource, that the status should equal P.

```
<simpleCondition>
 <variable name="Status"/>
       <operator name="="/>
       <value data="P"/>
   </simpleCondition>
```

In the above example, the <variable name="*value*"/> represents the attribute names recognized by the getGroupingAttributeValue (String attributeName, GroupContext context)() method on the resource. This method is part of the Groupable interface. For the purposes of Implicit Resource Group management within the WebSphere Commerce Administration Console, the attribute should also be defined in the ACATTR table and associated with the resource in the ACRESATREL table. When it is time to find the applicable policies for a given resource and action, this condition will be checked by calling the getGroupingAttributeValue(..) method, which in this case passes in Status as the attributeName parameter.

The <orListCondition>, specifies that the conditions within this block should be applied using a boolean OR. In this case, the status is either P or E. The<andListConditon>, specifies that the conditions within this block should be applied using a boolean AND. In this case, (Classname = com.ibm.commerce.order.objects.Order) AND (Status = P OR Status=E).

A sample attribute definition for populating the ACATTR table is shown in the following:

```
<Attribute Name="Status" Type="String">
</Attribute>
```

The Name element is a term to identify the attribute, and the Type element identifies the data type of the attribute. Possible values of the attribute are:

- String

- Integer
- Double
- Currency
- Decimal
- URL
- Image
- Date

The association of an attribute to a resource is specified within the Resource definition. For example, the `Status` attribute is associated with the `OrderResourceCategory` in the following example:

```
<ResourceCategory Name="com.ibm.commerce.order.objects.OrderResourceCategory"
     ResourceBeanClass="com.ibm.commerce.order.objects.Order" >

  <ResourceAttributes Name="Status"
   AttributeTableName="ORDERS"
   AttributeColumnName="STATUS"
   ResourceKeyColumnName="ORDERS_ID"/>
</ResourceCategory>
```

Where:

`<ResourceAttributes>`: A block of code that associates an attribute with a resource.

`AttributeTableName`: The name of the database table of the resource.

`AttributeColumnName`: The name of the column in the resource table that stores the attribute.

`ResourceKeyColumnName`: The name of the column in the resource table that stores the primary key.

Defining relationships

Access control policies have an optional relationship element. This relationship can only be created by loading an XML policy file with the relationship definition seen below:

```
  <Relation Name="value">
   </Relation>
```

The `Name` entry is the name of the relationship used in any policy, and is added to the `ACRELATION` table. `Name` corresponds to the relationship parameter of the `fulfills()` method on the protectable resource.

The following example displays the definition of a relationship called `creator`.

```
<Relation Name="creator">
</Relation>
```

Defining relationship groups

Relationship groups contain open conditions which are the conditions for belonging to the relationship group. If you need to define relationship groups, you must do so by defining the relationship group information in your XML file, or by modifying the `defaultAccessControlPolicies.xml` file as seen below:

```
<RelationGroup
  Name="aValue"
  OwnerID="Root Organization">
  <RelationCondition><![CDATA[
   <profile>
    Relationship Chain Open Condition XML
   </profile>
  ]]></RelationCondition>
 </RelationGroup>
```

Relationship chains

Each relationship group consists of one or more `RELATIONSHIP_CHAIN` open conditions, grouped by `andListCondition` or `orListCondition` elements. A relationship chain is a series of one or more relationships. The length of a relationship chain is determined by the number of relationships it consists of. This can be determined by examining the number of `<parmeter name= "X" value="Y">` entries in the XML representation of the relationship chain. The following is an example of a relationship chain with a length of one.

```
<openCondition name="RELATIONSHIP_CHAIN">
<parameter name="RELATIONSHIP"
value="aValue"/>
</openCondition>
```

Where:

`aValue`: A string representing the relationship between the user and the resource. This string should be one of the relationships checked in the fulfills method of the resource.

When a relationship chain has a length of two or more it is a series of two relationships. The first ,`<parmeter name= "X" value="Y">`, entry is between a user and an organizational entity. The last `<parmeter name= "X" value="Y">` entry is between an organizational entity and the resource. Intermediate `<parmeter name= "X" value="Y">` entries in the chain are between organizations. The following is an example of a relationship chain with a length of two.

```
<openCondition name=RELATIONSHIP_CHAIN">
<parameter name="aValue1" value="aValue2"/>
<parameter name="RELATIONSHIP" value="aValue3"/>
</openCondition>
```

Where:

`aValue1` : Possible values include `HIERARCHY` and `ROLE`. `HIERARCHY` specifies that there is a hierarchical relationship between the user and the organizational entity in the membership hierarchy. `ROLE` specifies that the user plays a role in the organizational entity. If the value of `aValue1` is HIERARCHY, the possible values of `aValue2` include `child`, which returns the organizational entity for which the user is a direct child in the member hierarchy. If the value of `aValue1` is ROLE, possible values of `aValue2` include any valid entries in the NAME column of the ROLE table which return all of the organizational entities for which the current user plays this role.

`aValue3`: A string representing the relationship between one or more organizational entities retrieved from evaluating the first parameter and the resource. This value corresponds to the relationship parameter of the `fulfills()` method on the protectable resource. If more than one organizational entity was returned by evaluating parameter aValue1 , this part of the RELATIONSHIP_CHAIN is satisfied if at least one of these organizational entities satisfies the relationship specified by parameter aValue2.

Note: For more information on defining relationship groups, see "Defining relationship groups" on page 146

Defining single-chain relationship groups

If as part of your access control policy, you are required to enforce that a user must belong to the organizational entity that is for example, the BuyingOrganizationalEntity of the resource, you will have to create a relationship group that is composed of one relationship chain that has a length of two. This is shown in the following example:

```
<RelationGroup Name="MemberOf->BuyerOrganizationEntity"
OwnerID="RootOrganization
<RelationCondition><![CDATA[
<profile>
<openCondition name="RELATIONSHIP_CHAIN">
<parameter name="HIERARCHY" value="child"/>
<parameter name="RELATIONSHIP" value="BuyingOrganizationalEntity"/>
</openCondition>
</profile>
]]><RelationCondition>
<RelationGroup>
```

The relationship chain has a length of two because it consists of two separate relationships. The first relationship is between the user and its parent organizational entity. The user is the child in that relationship. For the second relationship, the access control policy manager checks if the parent organizational entity fulfills the BuyingOrganizationalEntity relationship with the resource. In other words, it returns true if it is the buying organizational entity of the resource.

Note: For information on the openCondition tag, refer to the *WebSphere Commerce Accelerator Customization Guide*.

Another example would be if you had to enforce that the user have the role of Account Representative for the organizational entity that is the buying organizational entity of the resource. Again, this uses a relationship group that is composed of one relationship chain of a length of two. The first part of the chain finds all of the organizational entities for which the user has the Account Representative role. Then for the set of organizational entities, the access control policy manager checks if at least one of them fulfills the BuyingOrganizationalEntity relationship with the resource. If it does, a value of true is returned.

The following example shows how to define this type of relationship group:

```
<RelationGroup Name="AccountRep->BuyerOrganizationalEntity"
OwnerID="RootOrganization">
<RelationCondition><![CDATA[
<profile>
 <openCondition name="RELATIONSHIP_CHAIN">
 <parameter name="ROLE" value="Account Representative"/>
 <parameter name="RELATIONSHIP" value="BuyingOrganizationalEntity"/>
</openCondition>
 </profile>
]]><RelationCondition>
<RelationGroup>
```

Defining multiple-chain relationship groups

If you need to compose a relationship group that contains a multiple-chain relationship, you must specify whether the user must satisfy all of the relationship chains, meaning it is an AND scenario, or the user must satisfy at least one of the relationship chains, meaning it is an OR scenario.

In the following example the user must be the creator of the resource and must belong to the `BuyingOrganizationalEntity` specified in the resource. The first chain, that specifies the user must be the creator of the resource is has a length of one. The second chain, that specifies that the user must belong to the `BuyingOrganizationalEntity` specified in the resource, has a length of two.

```
<RelationshipGroup Name="Creator_And_MemberOf->BuyerOrganizationalEntity"
 OwnerID="RootOrganization">
<RelationCondition><![CDATA[
<profile>
<andListCondition>
<openCondition name="RELATIONSHIP_CHAIN">
<parameter name="RELATIONSHIP" value="creator" />
</openCondition>
<openCondition name="RELATIONSHIP_CHAIN">
<parameter name="HIERARCHY" value="child"/>
<parameter name="RELATIONSHIP" value="BuyingOrganizationalEntity"/>
</openCondition>
</andListCondition>
</profile>
]]></RelationCondition>
</RelationGroup>
```

Note: If you require the user to satisfy either of the two relationship chains, the `<andListConditon>` tag should be changed to the `<orListConditon>` tag.

Access groups

The default access groups that are part of WebSphere Commerce are found in language specific XML files, such as *WC_installdir*/xml/policies/xml/ACUserGroups_locale.xml. This file follows the DTD specified by *WC_installdir*/xml/policies/dtd/ACUserGroups_en_US.dtd.

The following is the format of an access group element:

```
<UserGroup Name="value"
   OwnerID="value"
   Description="value"

   <UserCondition>
     <![CDATA[
       <profile>
        Condition XML
       </profile>
     ]]>
   </UserCondition>
</UserGroup>
```

Where:

`Name`: The name of the access group, stored in the `MBRGRPNAME` column of the `MBRGRP` table.

`OwnerID`: The `Member ID` that owns this access group. The combination of `Name` and `OwnerID` must be unique. Special values that can be used include: `RootOrganization (-2001)` or `DefaultOrganization (-2000)`.

Description (optional): An optional attribute used to describe the access group.

UserCondition (optional): An optional element specifying implicit conditions of membership in this access group. This criteria is stored in the CONDITIONS column of the MBRGRPCOND table.

Condition XML: Using the condition framework, any valid combination of the orListCondition, andListCondition, simpleCondition, and trueConditionCondition elements.

The following SimpleCondition names are supported for the UserCondition element:

Table 13. Supported simple condition names

Variable Name	Description	Supported Operators	Supported Values	Qualifiers	Qualifier Values
role	Specifies that the user must have this role in the MBRROLE table..	= !=	Any value of the NAME column in the ROLE table.	org (if not specified, the user must have the role for any organization in the MBRROLE table.	• OrgEntityID : Where the user must have the role. • OrgAndAncestorOrgs: When it is used in a groupable template policy. This will check if the user has the specified role in the organization that owns the resource or any of its ancestor organizations.
registration status	Specifies that the user must have this registration status.	= !=	Any value of the REGISTER-TYPE column in the USERS table such as G for guest, and R for registered.	none	n/a
status	Specifies that the user must have this member state. This is usually used for the status of registration approval.	= !=	Any value of the STATE column in the MEMBER table such as 0 for pending registration approval, 1 for registration approved, and 2 for registration rejected.	none	n/a

Table 13. Supported simple condition names (continued)

Variable Name	Description	Supported Operators	Supported Values	Qualifiers	Qualifier Values
`org`	Specifies that the user is a child of the specified organization. This information is based on data stored in the MBRREL table	= !=	• Any value of the ORGENTITY_ID in the ORGENTITY table. • ?: if it is a groupable template policy. This will check if the user is a child of the organization that owns the resource. It will also check if the user is a child of any of the resource owner's ancestors, up to and including the closest ancestor that is subscribing to a policy group	none	n/a

Examples of simpleConditions for access groups

role:

Role without a qualifier: The following example displays a `role` simpleCondition without a qualifier; most commonly used in role-based policies. In this example the user must have a Seller Administration role for any organizational entity.

```
<UserConditon>
  <![CDATA[
  <profile>
   <simpleCondition>
    <variable name="role"/>
    <operator name="="/>
    <value data="Seller Administrator"/>
   </simpleCondition>
  </profile>
]]>
</UserCondition>
```

Role with a qualifier: The following example displays a `role` simpleCondition with a qualifier; most commonly used for organization-level policies. In this example the user must have a Seller role for the organizational entity with `ORGENTITY_ID = 100`.

```
<UserCondition>
   <!CDATA[
  <profile>
    <simpleCondition>
    <variable name="role"/>
    <operator name="="/>
    <value data="Seller"/>
     <qualifier name="org" data="100"/>
    <simpleCondition>
  </profile>
]]>
</UserCondition>
```

Role with a qualifier and parameter: The following example displays a role simpleCondition with a qualifier and the special data value OrgAndAncestorOrgs. This qualified data value, OrgAndAncestorOrgs, only works only in groupable template policies. In this example, the user must have a Sales Manager, Account Manager, or Seller role in the organization that owns the resource, or any of the organization's ancestors.

```
<UserCondition><!CDATA[
  <profile>
   <orListCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Sales Manager"/>
       <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Account Representative"/>
       <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Seller"/>
       <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
   </orListCondition>
  </profile/>
]]></UserCondition>
```

registrationStatus: The following example displays a `registrationStatus` simpleCondition. In this example, the user must be registered (`USERS.REGISTERTYPE = R`).

```
<UserCondition><!CDATA[
  <profile>
  <simpleCondition>
   <variable name="registrationStatus"/>
   <operator name="="/>
   <value data="R"/>
  </simpleCondition>
  </profile>
]]></UserCondition>
```

status: The following example displays a `status` simpleCondition. In this example, the user must have had registration approved. (`MEMBER.STATUS = 1`)

```
<UserCondition><![CDATA[
  <profile>
    <simpleCondition>
     <variable name="status"/>
     <operator name="="/>
     <value data="1"/>
     </simpleCondition>
  </profile>
]]></UserCondition>
```

org: The following example displays an `org` simpleCondition. In this example, the user must be registered in organizational entity `100`. In the MBRREL table, there must be a record where the user is a descendant of an organization that has `ANCESTOR_ID = 100`, and `SEQUENCE = 1`.

```
<UserCondition><![CDATA[
  <profile>
   <simpleCondition>
    <variable name="org"/>
    <operator name="="/>
    <value data="100"/>
   </simpleCondition>
  </profile>
]]>
</UserCondition>
```

Policies

The *WC_installdir*/xml/policies/xml/defaultAccessControlPolicies.xml file defines the default access control policies that are shipped out of the box. It follows the DTD specified by:
WC_installdir/xml/policies/dtd/accesscontrolpolicies.dtd.

The following is the template of a policy element:

```
<Policy Name="value"
  OwnerId="value"
  UserGroup="value"
  UserGroupOwner="value"
  ActionGroupName="value"
  ResourceGroupName="value"
  PolicyType="value"
  RelationName="value"
  RelationGroupName="value"
  RelationGroupOwner="value"
</Policy>
```

Where:

`Name`: The name of the policy. This is loaded into the `POLICYNAME` column of the `ACPOLICY` table. The `Name` and `OwnerID` together must be unique.

`OwnerID`: The member ID of the organizational entity that owns the policy. This will be loaded into the `member_id` column of the `ACPOLICY` table. The `OwnerID` and `Name` together must be unique. There are two special values that are recognized by the transformer tool, these are the `RootOrganization: -2001`, and `DefaultOrganization: -2000`

`UserGroup`: The name of the access group specified in the `MBRGRPNAME` column of the `MBRGRP` table. This is loaded into the `mbrgrp_id` column of the `ACPOLICY` table. The default access groups are defined in the *WC_installdir*/xml/policies/xml/ACUserGroups_language.xml file.

`UserGroupOwner`: The member ID of the member that owns the Access Group. This is needed when the access group is owned by a member other than the policy owner. If this is not specified, it is assumed that the access group is owned by the member that is specified by the `OwnerID` attribute.

`ActionGroupName`: The name of the action group specified in `GROUPNAME` column of `ACACTGRP` table. It is used to get the corresponding action group ID (`ACACTGRP_ID`) that will be stored in the `ACPOLICY` table. Role-based policies for controller commands have ActionGroupName set to ExecuteCommandActionGroup. Policies for databeans have ActionGroupName set to DisplayDatabeanActionGroup.

ResourceGroupName: The name of the Resource Group, specified in the GRPNAME column of the ACRESGRP table. It is used to get the corresponding resource group ID (ACRESGRP_ID) that is stored in the ACPOLICY table. Role-based policies for views have ResoureGroupName set to ViewCommandResourceGroup.

PolicyType: The type of policy. Valid values are groupableStandard and groupableTemplate. For backward compatibility, the values standard and template are also supported. If this attribute is unspecified when loading a new policy, the value null will be used. If this attribute is unspecified when updating an existing policy, the value will remain unchanged. The following table displays the mapping of string values to database values stored in the POLICYTYPE column of ACPOLICY table.

Table 14. Mapping of string values to database values

String	ACPOLICY.POLICYTYPE
groupableTemplate	3
groupableStandard	2
template	1
standard	0 or null

For more information on the types of policies see Chapter 3, “Authorization concepts,” on page 17.

RelationName (optional): The name of the Relationship, as specified in the RELATIONNAME column of the ACRELATION table. If it is specified, it is used to get the corresponding relationship ID (ACRELATION_ID) that is stored in the ACPOLICY table.

RelationGroupName (optional): The name of the Relationship Group, as specified in the GRPNAME column of the ACRELGRP table. If this attribute is specified, RelationName should not be specified, since Relationship Group takes precedence.

RelationGroupOwner: The member ID that owns the Relation Group. This attribute is necessary only if the RelationGroupName attribute is specified and if the value of the OwnerID attribute is not RootOrganization; in this case, RelationGroupOwner must be specified as RootOrganization (-2001).

Policy examples

Role-based policies:

For controller commands: In this example, users belonging to the AllUsers access group can execute controller commands that are part of the AllUserCmdResourceGroup resource group.

```
<Policy Name="AllUsersExceuteAllUserCmdResourceGroup"
   OwnerID="RootOrganization"
   UserGroup="AllUsers"
   ActionGroupName="ExecuteCommandActionGroup"
   ResourceGroupName="AllUserCmdResourceGroup"
   PolicyType="groupableStandard">
</Policy>
```

For views: In this example, users belonging to the MarketingManagers access group can execute the views belonging to MarketingManagersViews action group.

```
<Policy Name="MarketingManagersExecuteMarketingManagersViews"
   OwnerID="RootOrganization"
   UserGroup="MarketingManagers"
   ActionGroupName="MarketingManagersViews"
   ResourceGroupName="ViewCommandResourceGroup"
   PolicyType="groupableStandard">
</Policy>
```

Resource-level policies:

For commands: In this example, users belonging to `AllUsers` access group can perform the actions specified by the `CouponRedemption` action group on resources specified by the `CouponWalletResourceGroup`, as long as the users fulfill the `creator` relationship with respect to the resource.

```
<Policy Name="AllUsersExecuteCouponRedemptionCommandsOnCouponWalletResource"
   OwnerID="RootOrganization"
   UserGroup="AllUsers"
   ActionGroupName="CouponRedemption"
   ResourceGroupName="CouponWalletResourceGroup"
   RelationName="creator"
   PolicyType="groupableStandard">
</Policy>
```

For DataBeans: In this example, users belonging to the `AllUsers` access group can Display databeans specified by the `UserDatabeanResourceGroup` resource group, as long as the users fulfill the `owner` relationship with respect to the resource.

```
<Policy Name="AllUsersDisplayUserDatabeanResourceGroup"
   OwnerID="RootOrganization"
   UserGroup="AllUsers"
   ActionGroupName="DisplayDatabeanActionGroup"
   ResourceGroupName="UserDatabeanResourceGroup"
   RelationName="owner"
   PolicyType="groupableStandard">
</Policy>
```

Groupable template policies: In this example, users belonging to the

```
OrgAdminConsoleMembershipAdministratorsForOrg
```

access group, can perform the actions specified by the `ApproveGroupUpdate` action group on resources specified by the `OrganizationDataResourceGroup`.

```
<Policy Name="OrgAdminConsoleMembershipAdministratorsForOrgExecuteApprove
GroupUpdateCommandsOnOrganizationResource"
   OwnerID="RootOrganization"
   UserGroup="OrgAdminConsoleMembershipAdministratorsForOrg"
   ActionGroupName="ApproveGroupUpdate"
   ResourceGroupName="OrganizationDataResourceGroup"
   PolicyType="groupableTemplate">
</Policy>
```

Examining the definition of the `OrgAdminConsoleMembershipAdministratorsForOrg` access group would reveal the following condition for membership:

```
<UserCondition>
 <profile>
    <orListCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Buyer Administrator"/>
      <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Seller Administrator"/>
      <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
     <simpleCondition>
      <variable name="role"/>
      <operator name="="/>
      <value data="Channel Manager"/>
      <qualifier name="org" data="OrgAndAncestorOrgs"/>
     </simpleCondition>
    </orListCondition>
   </profile>
  UserCondition>
```

Note: The `simpleCondition` of `role` is qualified by `org = **OrgAndAncestorOrgs**`. `OrgAndAncestorOrgs` is a keyword that is only available in groupable template policies. It dynamically scopes the role to the context of the current resource's owner. In this example, the user must have one of the specified roles in the organization that owns the resource, or any of the organization's ancestors.

Defining policy groups

Policy groups are created to group policies, based on business and access control requirements. Some default policy groups are created out of the box; for more information see, "Default access control policies and groups," on page 201. Other policy groups are created, as needed, while publishing a store or a business model. In most cases you can simply add any new policies you create to existing policy groups. If you need to create a new policy group, you should define it in an XML file, similar to `defaultAccessControlPolicies.xml`, and then load it to the database. Here is a sample definition:

```
<PolicyGroup Name="aValue" OwnerID="aValue">
 </PolicyGroup>
```

where:

Name: The name of the policy group.

OwnerID: The member ID of the organizational entity that owns the policy group. This will be loaded into the `member_id` column of the `ACPOLGRP` table. The `OwnerID` and `Name` together must be unique. There are two special values that are recognized by the transformer tool, these are the `RootOrganization: -2001`, and `DefaultOrganization: -2000`.

Associating policies with policy groups

Policies can belong to multiple policy groups. However, to ease administration of policies, it is recommended that a policy belong to only one policy group. This association should be defined in an XML file, similar to `defaultAccessControlPolicies.xml`, and then loaded to the database. Here is a sample definition:

```
<PolicyGroup Name="aValue" OwnerID="aValue">
  <PolicyGroupPolicy  Name="aValue" PolicyOwnerID="aValue" />
</PolicyGroup>
```

where:

PolicyGroupPolicy Name: The name of the policy, previously defined, to be associated with the specified policy group. This policy must have one of the following policy types: groupableStandard or groupableTemplate.

PolicyGroupPolicy PolicyOwnerID (optional): The member ID of the organizational entity that owns the specified policy. If this parameter is not specified, the default value is `OwnerID` of the policy group. There are two special values that are recognized by the transformer tool, these are the `RootOrganization: -2001`, and `DefaultOrganization: -2000`.

Subscribing to policy groups

The resources of an organization are protected by the policies in the policy groups to which that organization subscribes. If that organization does not subscribe to any policy groups, then the policy groups to which that organization's closest ancestor subscribes will apply. For more information about which policy groups an organization should subscribe to, see "Default access control policies and groups," on page 201.

Policy group subscription can be done in the Organization Administration Console, but can also be defined in an XML file, similar to `defaultAccessControlPolicies.xml`, and then loaded to the database. Here is a sample definition:

```
<PolicyGroup Name="aValue" OwnerID="aValue">
  <PolicyGroupSubscription OrganizationID="aValue"/>
 </PolicyGroup>
```

where:

OrganizationID: The member ID of the organizational entity that is subscribing to this policy group. There are two special values that are recognized by the transformer tool, these are the `RootOrganization: -2001`, and `DefaultOrganization: -2000`.

Translatable policy data

The following is a template of a customized policy file that can be used to define translatable policy data:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
 <!--The following TRANSLATABLE access control elements should
 be defined in this file:
 <Attribute_nls>
 <Action_nls>
 <Relation_nls>
 <ResourceCategory_nls>
 <ActionGroup_nls>
 <ResourceGroup_nls>
 <Policy_nls>
 <PolicyGroup_nls>-->
<!DOCTYPE PoliciesNLS SYSTEM "../dtd/accesscontrolpoliciesnls.dtd">

<PoliciesNLS LanguageID="value">


 <!--Insert access control element definitions here -->
 </PoliciesNLS>
```

The `LanguageID` attribute is a string corresponding to the language of the locale-specific data. Valid values of the `LanguageID` are:

- en_US
- fr_FR
- de_DE

- it_IT
- es_ES
- pt_BR
- zh_CN
- zh_TW
- ko_KR
- ja_JP

Non-translatable policy data

The following is a template of a customized policy file containing non-translatable data:

```
  <?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>

<!DOCTYPE Policies SYSTEM "../dtd/accesscontrolpolicies.dtd">

 <!--The following NON-TRANSLATABLE access control elements
 should be defined in this file:

 <Attribute>
 <Action>
 <ResourceCategory>
 <Relation>
 <RelationGroup>
 <ActionGroup>
 <ResourceGroup>
 <Policy>
 <PolicyGroup>-->
<Policies>

 <!--Insert access control element definitions here-->
 </Policies>
```

Locale-specific data

The following optional locale-specific data can be loaded to give additional description to the access control elements already defined in the non-translatable XML file. The default locale-specific data can be found at the following address:

`WC_installdir\xml\policies\xml\`
`defaultAccessControlPolicies_locale.xml`

For example, `defaultAccessControlPolicies_en_US.xml`.

Attribute: The following example defines additional attribute element information:

```
<Atrribute_nls AttributeName="Status"
DisplayName_nls="Status attribute"
Description_nls="Resource status attribute"
/>
```

Where:

`AttributeName`: The name of the attribute. This value is stored in the `ATTRNAME` column of the `ACATTR` table.

`DisplayName_nls`: The display name of the attribute. This value is stored in the `DISPLAYNAME` column of the `ACATTRDESC` table.

`Description_nls`: An optional description of the attribute. This value is stored in the DESCRIPTION column of the `ACATTRDESC` table.

Action: The following example defines additional action element information:

```
<Action_nls ActionName="OrderAdjustmentButton"
DisplayName_nls="Order Adjustment Button View"
Description_nls="The view for loading buttons in the order adjustment page
 when placing an order from Commerce Acclerator"
/>
```

Where:

`ActionName`: The name of the action. This value is stored in the ACTION column of the `ACACTION` table.

`DisplayName_nls`: The display name of the action. This value is stored in the `DISPLAYNAME` column of the `ACACTDESC` table.

`Description_nls`: An optional description of the action. This value is stored in the DESCRIPTION column of the `ACACTDESC` table.

Relation: The following example defines additional relation element information:

```
<Relation_nls RelationName="creator"
DisplayName_nls="creator"
Description_nls="creator"
/>
```

Where:

`RelationName`: The name of the relationship. This value is stored in the `RELATIONNAME` column of the `ACRELATION` table.

`DisplayName_nls`: The display name of the relationship. This value is stored in the `DISPLAYNAME` column of the `ACRELDESC` table.

`Description_nls`: An optional description of the relationship. This value is stored in the DESCRIPTION column of the `ACRELDESC` table.

Resource Category: The following example defines additional resource category information:

```
<ResourceCategory_nls ResourceCategoryName="com.ibm.commerce.
catalog.objects."InterestItemList"
DisplayName_nls="Interest Item List"
Description_nls="Interest Item List command"
/>
```

Where:

`ResourceCategoryName`: The name of the resource category. This value is stored in the `RESCLASSNAME` column of the `ACRESCGRY` table.

`DisplayName_nls`: The display name of the resource category. This value is stored in the `DISPLAYNAME` column of the `ACRSCGDES` table.

`Description_nls`An optional description of the resource category. This value is stored in the DESCRIPTION column of the `ACRSCGDES` table.

Action Group: The following example defines additional action group information:

```
<ActionGroup_nls ActionGroupName="DoEverything"
DisplayName_nls="Do Everything"
Description_nls="Permits access to all Actions"
/>
```

Where:

ActionGroupName: The name of the action group. This value is stored in the GROUPNAME column of the ACACTGRP table.

DisplayName_nls: The display name of the action group. This value is stored in the DISPLAYNAME column of the ACACGPDESC table.

Description_nls: An optional description of the action group. This value is stored in the DESCRIPTION column of the ACACGPDESC table.

Resource Group: The following example defines additional resource group information:

```
<ResourceGroup_nls ResourceGroupName="AllResourceGroup"
DisplayName_nls="All Resources Group"
Description_nls="All Resources"
/>
```

Where:

ResourceGroupName: The name of the resource group. This value is stored in the GRPNAME column of the ACRESGRP table.

DisplayName_nls: The display name of the resource group. This value is stored in the DISPLAYNAME column of the ACRESGPDES table.

Description_nls: An optional description of the resource group. This value is stored in the DESCRIPTION column of the ACRESGPDES table.

Policy: The following example defines additional policy information:

```
<Policy_nls PolicyName="SiteAdministratorsCanDoEverything"
OwnerID="RootOrganization"
DisplayName_nls="Site Administrators Can Do Everything"
Description_nls="Policy that allows Site Administrators to do everything"
/>
```

Where:

PolicyName: The name of the access control policy. This value is stored in the POLICYNAME column of the ACPOLICY table.

OwnerID: The member ID of the organizational entity that owns this policy.

DisplayName_nls: The display name of the policy. This value is stored in the DISPLAYNAME column of the ACPOLDESC table.

Description_nls: An optional description of the policy. This value is stored in the DESCRIPTION column of the ACPOLDESC table.

Policy group: The following example defines additional policy group information:

```
<PolicyGroup_nls PolicyGroupName="B2CPolicyGroup" OwnerID="RootOrganization"
  DisplayName_nls="B2C Policy Group"
  Description_nls="This policy group contains all the B2C specific policies."
 />
```

where:

`PolicyGroupName`: The name of the access control policy group to which additional information is being added. This value is found in the NAME column of the `ACPOLGRP` table.

`OwnerID`: The member ID of the organizational entity that owns this policy group.

`DisplayName_nls`: The display name of the policy group. This value is stored in the `DISPLAYNAME` column of the `ACPLGPDESC` table.

`Description_nls`: An optional description of the policy group. This value is stored in the DESCRIPTION column of the `ACPLGPDESC` table.

After you have changed the XML files

Testing your changes

For information on testing your changes see "After you make your policy changes" on page 97.

Loading your changes into the database

If you make policy changes by working directly with the XML files, you must load the changed XML files back into the databases. It is important to maintain consistency between the XML files and the access control information in the databases for several reasons:

- When you create an instance of WebSphere Commerce, the policy and access group definitions are loaded from the XML files.
- If you want to implement the same access control policies in a second instance of WebSphere Commerce, you can do so by copying the XML files to the appropriate directory before creating the second instance.
- The XML files offer a convenient way to directly view and edit your policies and their component parts so keeping the files up-to-date is essential.

Loading your XML changes into the database

The loading process reads the XML files containing the access control policy information and access group definitions and loads them into the appropriate databases. The policy and access group information contained in the XML files is loaded at installation, however, you must reload the files if you make changes to them.

Notes:

1. If you create customized XML files, you need to copy them into the `<`*`WC_installdir`*`>/xml/policies/xml` directory to have them loaded into the databases.
2. There is a setting in the loading scripts that specifies the following parameter setting while resolving ID's and loading the data to the database: `"-maxerror 100000"`. This means that if there up to 100000 foreign key violations while loading the data, they will be ignored, instead of aborting. This value can be

increased or decreased as needed. For example, if you want to stop after one such error, you would change the value to 1.

For ▶ 400 : if you create customized XML files, you must use the full path to the DTD in your file. The access control policies DTDs are located in `WC_installdir/xml/policies/dtd`.

To load the access groups and access control policies, run the following commands.

For ▶ 2000

1. From the directory `<WC_installdir>\bin`, run the following command files as needed in the order listed here:
   - To load the user (access) group definitions, run the **acugload** command file. **Syntax**: acugload.cmd *<database name> <database user> <database user password> <UserGroups xml file>*[*schema name*] **Example**: acugload mall dbuser dbusrpwd ACUserGroups_en_US.xml
   - To load the main access control policies file, run the **acpload** command file. **Syntax:** acpload.cmd *<database name> <database user> <database user password> <Policies xml file>*[*schema name*] **Example**: acpload mall dbuser dbusrpwd defaultAccessControlPolicies.xml
   - To load the display names and descriptions file, run the **acpnlsload** command file. **Syntax:** acpnlsload.cmd *<database name> <database user> <database user password> <NLS Policies xml file>*[*schema name*] **Example**: acpnlsload mall dbuser dbusrpwd defaultaccesscontrolpolicies_en_US.xml
2. Check the log files **acugload.log**, **acpload.log,** and **acpnlsload.log** in `<WC_installdir>\logs` for any errors.

For ▶ 400 ▶ AIX ▶ Solaris ▶ Linux

The database user ID must have the following permission in order to proceed with the following steps:

- read/write/execute authority to the directories, subdirectories, and files of `WC_installdir/xml/policies` and `WC_installdir/logs`.
- read/execute authority to the `WC_installdir/bin` directory and its files.

If the database user ID does not have the above required authority, you need to grant this authority using the `chmod` command.

1. Login as the database user ID.
2. From the directory `<WC_installdir>/bin`, run the following shell scripts as needed in the order listed here:

1. To load the user (access) group definitions, run the **acugload** shell script. **Syntax:** acugload.sh *<database name> <database user> <database user password> <UserGroups xml filename>*[*schema name*] **Example**: acugload mall dbuser dbusrpwd ACUserGroups_en_US.xml
2. To load the main access control policies file, run the **acpload** shell script. **Syntax:** acpload.sh *<database name> <database user> <database user password> <Policies xml filename>*[*schema name*] **Example**: acpload mall dbuser dbusrpwd defaultAccessControlPolicies.xml
3. To load the display names and descriptions file, run the `acpnlsload` shell script. **Syntax:** acpnlsload.sh *<database name> <database user> <database user password> <NLS Policies xml filename>*[*schema name*] **Example**: acpnlsload mall dbuser dbusrpwd defaultaccesscontrolpolicies_en_US.xml

Check the log files `acugload.log`, `acpload.log`, and `acpnlsload.log` in `<WC_installdir>/logs` for any errors.

Note: After performing these scripts you must check the log files, as any errors that may occur while running these scripts will not appear on the command line.

For 400

Note: For 400 the log files are located in `WC_userdir/instances`.

Extracting policy and access group definitions from the databases into your XML files

The extraction process reads the policy and access group information in the access control databases and generates files that capture the information in XML format. The extraction utility uses an input filter XML file to specify which data to extract from the database. The following filter files are provided:

- `ACPoliciesfilter.xml`: used to extract all access group and policy data.
- `ACUserGroupsFilter.xml`: used to extract all access group data.
- `OrganizationPoliciesFilter.xml`: used to extract all access group and policy data for a particular organization. Before using this file, it should be edited to specify the required organization ID. The policy data owned by this organization ID will be extracted.

For NT 2000

1. From the `<WC_installdir>\bin` directory, run the following `acpextract` command:

   ```
   acpextract.cmd <database name> <database user> <database user password>
   <input xml filter file> [schema name]
   ```

   For example,

   ```
   acpextract.cmd mall dbuser dbusrpwd ACPoliciesfilter.xml
   ```

   The following files are created:

   - `ExtractedACPolicies.xml`: This file contains data extracted by the `Extract` command for the given filter criteria.
   - `ExtractedACPolicies.dtd`: The DTD for the `ExtractedACPolicies.xml` file.
   - `AccessControlUserGroups.xml`: The file containing the access group definitions.
   - `AccessControlPolicies.xml`: The file containing the language-independent access control policy information.
   - `AccessControlPolicies_LOCALE.xml`: The language-dependent access control policies file that contains the display names and descriptions.
2. Check the log file `<WC_installdir>\logs\acpextract.log` for any processing errors that might have occurred.

For 400 AIX Solaris Linux

1. Login as the database user ID.
2. From the `<WC_install_dir>\bin` directory, run the following `acpextract` shell script:

   ```
   acpextract.sh <database name> <database user>
   <database user password> <input xml filter file> [schema name]
   ```

For example,

```
acpextract.sh mall dbuser dbusrpwd ACPoliciesfilter.xml
```

The following files are created:

- `ExtractedACPolicies.xml`: This file contains data extracted by the `Extract` command for the given filter criteria.
- `ExtractedACPolicies.dtd`: The DTD for the `ExtractedACPolicies.xml` file.
- `AccessControlUserGroups.xml`: The file containing the access group definitions.
- `AccessControlPolicies.xml`: The file containing the language-independent access control policy information.
- `AccessControlPolicies_LOCALE.xml`: The language-dependent access control policies file that contains the display names and descriptions.

3. Check the log file `<`*`WC_installdir`*`>\logs\acpextract.log` for any processing errors that might have occurred.

For ▶ 400

1. The following files are created in the *`WC_installdir`*`/xml/policies/xml` directory using the `OUTDIR` parameter:
   - `ExtractedACPolicies.xml`: This file contains data extracted by the `Extract` command for the given filter criteria.
   - `ExtractedACPolicies.dtd`: The DTD for the `ExtractedACPolicies.xml` file.
   - `AccessControlUserGroups.xml`: The file containing the access group definitions.
   - `AccessControlPolicies.xml`: The file containing the language-independent access control policy information.
   - `AccessControlPolicies_LOCALE.xml`: The language-dependent access control policies file that contains the display names and descriptions.

Part 4. Payments security

This part describes the Payments security administration tasks.

Chapter 14. WebSphere Commerce Payments access

WebSphere Commerce Payments authenticates users through the use of realms. A realm is a registry of users along with a single method of authenticating those users, for example, a user's name and password. Each WebSphere Commerce Payments installation can only use one realm at a time. Examples of realm types include LDAP realms and operating system realms. A user must be defined in a realm before being granted access to resources. Therefore, a user is a valid WebSphere Commerce Payments user if, and only if, they are both:

- In the realm
- Assigned a role in WebSphere Commerce Payments

WebSphere Commerce Payments employs a role-based access control scheme which defines four WebSphere Commerce Payments roles:

1. Payments Administrator
2. Merchant Administrator
3. Supervisor
4. Clerk

The Payments Administrator can use the WebSphere Commerce Payments user interface User window to assign access (based on role) to a user defined in a realm. The WCSRealm is provided with WebSphere Commerce Payments. The WCSRealm class is automatically configured for your system. This realm allows the WebSphere Commerce Payments Servlet to use the administrator information that is already registered in the WebSphere Commerce user tables. This administrator information is used for Payments Administrators so that you do not have to define another set of administrator IDs to use the WebSphere Commerce Payments user interface.

Chapter 15. Maintaining WebSphere Commerce Payments security

WebSphere Commerce Payments security is built on several key security elements. These elements combine to create an environment in which services can be deployed securely on the Web.

Note: IBM WebSphere Commerce Payments (hereafter called WebSphere Commerce Payments) was previously known as Payment Manager. Starting with version 3.1.3, the payments application was renamed to WebSphere Commerce Payments and references to the product were changed throughout this document.

Protecting WebSphere Commerce Payments

In the heart of WebSphere Commerce Payments is the Payment Servlet. Several ancillary products, the Web server configured with WebSphere Application Server, the database, and the user interface, complete the WebSphere Commerce Payments picture. This chapter discusses methods for securing the various WebSphere Commerce Payments components.

Protecting sensitive data

For each query command, the framework verifies the user's role against that minimum role and thereby, sets an indicator in the QueryRequest object to indicate whether sensitive data such as credit card numbers or billing addresses should be returned in full view or if it should be masked out. The WebSphere Commerce Payments framework does not maintain any sensitive data that can be returned via a query command. However, new methods are provided to cassette writers to check the value of this indicator and also to mask sensitive data in a standardized way. Each cassette must discern the sensitive data from the rest of the stored data. Typically, the sensitive data is the same set of data that a cassette encrypts before storing it to the WebSphere Commerce Payments database.

The JVM system parameter `wpm.MinSensitiveAccessRole={clerk|supervisor|madmin|psadmin|none}` specifies the minimum role a user must have to be allowed access to sensitive data. The value is case-sensitive. If this property is not specified, a value of clerk is assumed, allowing all users to see sensitive data. If an invalid value is specified, the Payment Servlet fails to initialize.

Note that this parameter can be set during Payments instance creation and updated at any time using the WebSphere Commerce Configuration Manager. The name of the parameter in Configuration Manager is Minimum Access Role in the Payments instance panel. For more information on the Configuration Manager panels, see the *WebSphere Commerce Installation Guide* for your platform, or see the online help for the Payments instance panel while in Configuration manager.

The following table describes supported values, which are listed in increasing order of authority:

Table 15. Payments user role authority

User	Description

Table 15. Payments user role authority (continued)

clerk	Users with a role of clerk or higher can see sensitive data.
supervisor	Users with a role of supervisor or higher can see sensitive data.
madmin	Users with a role of Merchant Administrator or higher can see sensitive data.
psadmin	Only Payments Administrators can see sensitive data.
none	No one is allowed to see sensitive data.

You can specify the `wpm.MinSensitiveAccessRole` parameter through the WebSphere Commerce Configuration Manager.

Protecting the database

The WebSphere Commerce Payments database stores sensitive data and requires protection from reading and writing by unauthorized sources. WebSphere Commerce Payments provides support for the encryption of sensitive data – for example, passwords and cardholder information – that is stored in the database.

Transaction data

Below are some guidelines for handling transaction data.

- Sensitive transactional information is stored in a database table in the instance library. This library is specified as the Instance Schema Name in the Payments Instance Creation Wizard.
- Any backups should be kept secure.
- The database tables in the instance library contain critical configuration and transaction information and should be included as part of your system backup strategy. You should also back up the following:
  - Files in the `/QIBM/UserData/CommercePayments/V55/`*`instance`* directory where Instance is the name of the WebSphere Commerce Payments instance
  - HTTP server instance that you configured for WebSphere Commerce Payments. This HTTP server is specified as the Web Server in the Payments Instance Creation Wizard.
  - Objects in the instance library on the local machine as well as the database collection on the remote machine when remote database storage is used.

Part 5. Miscellaneous security topics

This part describes miscellaneous security tasks that can be performed by the WebSphere Commerce system administrator.

Chapter 16. Enabling WebSphere Application Server security

This chapter describes how to enable security for WebSphere Application Server. Enabling WebSphere Application Server security prevents all Enterprise JavaBeans components from being exposed to remote invocation by anyone.

Important note on WebSphere Application Server default certificates

| WebSphere Application Server security and WebSphere Commerce Payments
| are configured to use the `DummyServerKeyFile.jks` and
| `DummyServerTrustFile.jks` files with the default self-signed certificate
| *out-of-the-box*. Using the dummy key and trust file certificates is not safe and
| consequently you should generate your own certificate to replace the dummy
| certificates immediately. Please refer to the WebSphere Application Server
| Security Guide for more information on the dummy key and trust file
| certificates and how to replace them. You can access this Guide at:
| `ftp://ftp.software.ibm.com/software/webserver/appserv/library/wasv502base_sec.pdf`

Notes:

1. Windows If WebSphere Application Server global security is enabled as outlined in the steps in this chapter, you will not be able to stop theWebSphere Application Server server (for example, `server1`) properly from the Windows 2000 Services panel. To stop the service when security is enabled, use the `stopserver` command from the `WAS_installdir\bin` directory in a command prompt as follows:

   ```
   stopserver server -username user_id -password password
   ```

   where *server* is the name of the WebSphere Application Server configuration directory of the server you want to stop (for example, `server1`), *user_id* is the user name for authentication, if security is enabled in the server, and *password* is the the password for authentication, if security is enabled in the server.

   When you attempt to stop the server from the Services panel, the properties are such that the user ID and password are not included. With global security enabled, both the user ID and password are required for authentication when you stop the server. The service continues to run (despite the Services panel showing that it has stopped). Note that the user ID and password are not required to start the service from the Services panel.
2. If you need to stop the application server when WebSphere Application Server security is enabled, use the `stopserver` command from the `WAS_installdir/bin` directory in a command prompt as follows:

   AIX Linux Solaris

   ```
   | ./stopserver.sh server -username user_id -password password
   ```

   where *server* is the name of the WebSphere Application Server application server you want to stop (for example, `server1`), *user_id* is the user name for authentication, and *password* is the the password for authentication.

   400

   ```
   stopserver -instance WAS_instancename server -username user_id
      -password password
   ```

where *WAS_instancename* is the name of the WebSphere Application Server instance, *server* is the name of the WebSphere Application Server application server you want to stop (for example, `server1`), *user_id* is the user name for authentication, and *password* is the the password for authentication.

3. AIX Linux Solaris Windows When enabling WebSphere Application Server security, it is strongly recommended that your machine meets the following requirements:
   - A minimum machine memory of 1 GB.
   - A minimum heap size of 384 MB, for the WebSphere Commerce application.
4. Also, with WebSphere Application Server security enabled, the `stopNode` (`stopNode.sh` on AIX, Linux, or Sun) command for the node agent will also require that you specify a user name and password.

Before you begin

Before you begin to enable security, you will need to know how the WebSphere Application Server where you are enabling security validates user IDs. WebSphere Application Server can use either LDAP or the operating system's user registry as the WebSphere Application Server user registry.

Enabling security with an LDAP user registry

AIX Solaris Linux To enable WebSphere Application Server security when you are using LDAP as the WebSphere Application Server user registry, log into the system as the `wasuser` ID and perform the following steps.

400 To enable WebSphere Application Server security when you are using LDAP as the WebSphere Application Server user registry, log into the system, and perform the following steps.

Windows To enable WebSphere Application Server security when you are using LDAP as the WebSphere Application Server user registry, log into the system as a user with administrative authority, and perform the following steps.

1. Start WebSphere Application Server and open the WebSphere Application Server Administration Console.
2. In the Administration Console, modify the global security settings as follows:
   a. Under **Security**, expand **User Registries** and click **LDAP**. Fill in the fields in the **Configuration** tab as follows, depending on the type of directory server you are using:

AIX 400 Linux Solaris Windows

Table 16. IBM Directory Server Users.

Field Name	Definition	Sample Values	Notes
Server User ID	User ID	*user_ID*	• This must not be the LDAP administrator. • Do not use a user that has been specified as cn=xxx. • Ensure that the object class of this user is compatible with the object class specified in the User Filter field of the LDAP Advanced Properties window.
Server User password	User Password	*password*	
Type	Type of LDAP server	SecureWay	
Host	Host name of the LDAP server	*hostname.domain.com*	
Port	Port that the LDAP server is using		This field is not required
Base Distinguished Name	Distinguished Name under which searching occurs	o=ibm,c=us	
Bind Distinguished Name	Distinguished Name for binding to the directory when searching		This field is not required
Bind Password	Password for the Bind Distinguished Name		This field is not required

Table 17. Netscape Users. ▶ Windows

Field Name	**Definition**	**Sample Values**	**Notes**
Server User ID	User ID	*user_ID*	• This must not be the LDAP administrator. • Do not use a user that has been specified as cn=xxx. • Ensure that the object class of this user is compatible with the object class specified in the User Filter field of the LDAP Advanced Properties window.
Server User Password	User Password	*password*	
Type	Type of LDAP server	Netscape	
Host	Host name of the LDAP server	*hostname.domain.com*	
Port	Port that the LDAP server is using		This field is not required
Base Distinguished Name	Distinguished Name under which searching occurs	o=ibm	
Bind Distinguished Name	Distinguished Name for binding to the directory when searching		This field is not required
Bind Password	Password for the Bind Distinguished Name		This field is not required

Table 18. Domino Users. ▶ Windows

Field Name	**Definition**	**Sample Values**	**Notes**
Server User ID	Short Name/User ID	*user_ID*	Ensure that the object class of this user is compatible with the object class specified in the User Filter field of the LDAP Advanced Properties window.
Server User Password	User Password	*password*	
Type	Type of LDAP server	Domino™ 5.0	
Host	Host name of the LDAP server	*hostname.domain.com*	

Table 18. Domino Users (continued). Windows

Field Name	Definition	Sample Values	Notes
Port	Port that the LDAP server is using		This field is not required
Base Distinguished Name	Distinguished Name under which searching occurs		This field is not required
Bind Distinguished Name	Distinguished Name for binding to the directory when searching		This field is not required
Bind Password	Password for the Bind Distinguished Name		This field is not required

Table 19. Active Directory Users. Windows

Field Name	Definition	Sample Values	Notes
Server User ID	sAMAccountName	*user_ID*	• User Logon Name of any ordinary user. • Do not use a user that has been specified as cn=xxx. • Ensure that the object class of this user is compatible with the object class specified in the User Filter field of the LDAP Advanced Properties window.
Server User Password	User Password	*password*	
Type	Type of LDAP server	Active Directory	
Host	Host name of the LDAP server	*hostname.domain.com*	
Port	Port that the LDAP server is using		This field is not required
Base Distinguished Name	Distinguished Name under which searching occurs	CN=users, DC=domain1, DC=domain2, DC=com	
Bind Distinguished Name	Distinguished Name for binding to the directory when searching	CN=*user_ID*, CN=users, DC=domain1, DC=domain2, DC=com	The *user_ID* value is the Display Name. This is not necessarily the same as the User Logon Name.

Table 19. Active Directory Users (continued). Windows

Field Name	Definition	Sample Values	Notes
Bind Password	Password for the Bind Distinguished Name	*bind_password*	This should be the same as the Security Server Password.

Click **Apply**.

b. In the Administration Console, expand **Security**, then expand **Authentication Mechanisms** and click **LTPA**.
   1) In the LPTA Configuration tab, fill in the LTPA settings as required and click **Apply**.
   2) Under Additional Properties click **Single Signon (SSO)** and clear the **Enabled** check box if you do not want to use this functionality.
   3) Click **Apply**.

c. In the Administration Console, expand **Security** and click **Global Security**.
   1) In the Global Security Configuration tab, select **Enabled** and clear **Enforce Java 2 Security**.

      Note: WebSphere Commerce 5.5 does not support Java 2 security.
   2) In the Active Authentication Mechanism field, select **Lightweight Third Party Authentication (LTPA)**.
   3) In the Active User Registry field, select **LDAP**.
   4) Click **Apply**.

d. In the Administration Console, expand **Applications**, then click **Enterprise Applications**.
   1) In the Enterprise Applications window, click your Commerce application, **WC_***instance_name* (for example, `WC_demo`).
   2) Under Additional Properties, click **Map security roles to users/groups**.
   3) Select `WCSecurityRole` using the check box at the left and click **Lookup users** and locate the user whose role you wish to map.

      The following are example steps to look up an LDAP user and map the `WCSecurityRole` role to that user. These steps are specific to WebSphere Application Server Network Deployment for an LDAP user named `myuser`. The steps on your system should be similar but could vary slightly:
      a) Using a search string of "*", click **Search**.
      b) In the Available panel, the `myuser` distinguished name (for example, `uid=myuser,cn=users,dc=ibm,dc=com`) should be retrieved from the LDAP server. Select it and click the >> button to move it into the Selected panel.
      c) Click **OK**.
      d) Click **OK** again in the "Mapping Users to Roles" panel.
      e) If the Dynamic Cache Monitor is installed, repeat this process to also assign the `Administrator` role to the `myuser` user.
      f) Click **Save**.
      g) If you are using WebSphere Application Server Network Deployment select the **Synchronize changes with Nodes** check box.
      h) Click **Save** again to apply the changes to the master configuration.

3. Close the Administration Console, and stop all active application servers. From now on, when you open the WebSphere Application Server Administration

Console, you will be prompted for the Security Server ID and password. Do not restart the WebSphere Commerce server yet, since you still need to configure security in the WebSphere Commerce Configuration Manager.

If you are running WebSphere Application Server Network Deployment, then you must also stop the node agents, and the deployment manager.

4. Open the WebSphere Commerce Configuration Manager and select **Commerce** > **Instance List** > *instance_name* **> Instance Properties > Security** and click the **Enable Security** check box. Select LDAP User Registry for the Authentication Mode. You are prompted to enter the user name that you selected in step 2d3 on page 178 and its associated password. Click **Apply**, then exit Configuration Manager.
5. Restart all application servers.

Enabling security with an operating system user registry

AIX Linux Solaris To use the operating system as a user registry, WebSphere Application Server needs to be run as the `root` ID. Run WebSphere Application Server as `root` and perform the following steps.

400 Windows To enable WebSphere Application Server security when you are using the operating system user validation as the WebSphere Application Server user registry, log in as a user with administrative authority and perform the following steps.

1. AIX Linux Solaris Login as root.
2. AIX Linux Solaris Start the WebSphere Application Server and launch the WebSphere Application Server Administration Console while logged in as `root`. To startup the server:

   ```
   cd WAS_installdir/bin
   ./startServer server
   ```

   where *server* is the name of the WebSphere Application Server application server, for example, `server1`.
3. In the WebSphere Application Server Administration Console, modify the global security settings as follows:
   a. In the Administration Console, expand **Security**, expand **User Registries** and click **Local OS**. Fill in the fields in the **Configuration** tab as follows, for your security registry server:

Field Name	Sample Values	Notes
Server User ID	*wcsuser*	400 The user ID on iSeries should have the `*SECOFR` authority. AIX Solaris Linux A user ID that is root or has root authority. Windows The user ID with operating system administrative privileges that you logged in with. If the machine belongs to a domain, use the fully-qualified user ID. For example: *DomainXYZ\user_id*. Ensure that this account exists in the domain server and is a member of the Administrator's group.
Security Server Password	*password*	This is the password belonging to the user with operating system administrative privileges that you logged in with.

Click **Apply** and then **Save**.

b. In the Administration Console, expand **Security** and click **Global Security**.
   1) In the Global Security Configuration tab, select **Enabled** and clear **Enforce Java 2 Security**.
   2) In the Active Authentication Mechanism field, select **SWAM (Simple WebSphere Authentication Mechanism**.
   3) In the Active User Registry field, select **Local OS**.
   4) Click **Apply** and then **Save**.

4. In the Administration Console, expand **Applications**, then click **Enterprise Applications**.
   a. In the Enterprise Applications window, click your Commerce application, **WC_***instance_name* (for example, `WC_demo`).
   b. Under Additional Properties, click **Map security roles to users/groups**.
   c. Click **Lookup users** and locate the user whose role you wish to map.
   d. For that user, select the **WCSecurityRole** and click **OK**.
5. Open the WebSphere Commerce Configuration Manager and select **Instances List** → *`instance_name`* → **Instance Properties**→ **Security** and select the **Enable Security** check box. Select **Operating System User Registry** for the authentication mode, and to enter the user name and password that you entered in step 3a on page 179. Click **Apply** then exit Configuration Manager.
6. Stop and restart the WebSphere Application Server administration server. From now on, when you open the WebSphere Application Server Administration Console, you will be prompted for the Security Server ID and password.

Disabling WebSphere Commerce EJB security

WebSphere Commerce Business Edition allows you to disable EJB security. To disable WebSphere Commerce EJB Security, do the following:

1. Start the WebSphere Application Server Administration Console.
2. In the Administration Console, expand **Security** and click **Global Security**. In the Global Security Configuration tab, clear the **Enabled** check box.
3. Open the WebSphere Commerce Configuration Manager and select**Instances List** → *instance_name* → **Instance Properties**→ **Security** and clear the **Enable Security** check box.
4. Exit the WebSphere Application Server Administration Console.
5. Stop and restart the WebSphere Application Server administration server.

WebSphere Commerce security deployment options

WebSphere Commerce supports various security deployment configurations. The following table illustrates the security deployment options available to you.

Table 20. Single machine security scenarios

WebSphere Application Server security is enabled.	• Use the operating system as the WebSphere Application Server registry. • Use the database as the WebSphere Commerce registry.
	• Use LDAP as the WebSphere Application Server registry. • Use LDAP as the WebSphere Commerce registry.
	• Use LDAP as the WebSphere Application Server registry.
WebSphere Application Server security is disabled, and your WebSphere Commerce site is located behind a firewall.	• A WebSphere Application Server registry is not required. • Use the database as the WebSphere Commerce registry.
	• A WebSphere Application Server registry is not required. • Use LDAP the WebSphere Commerce registry.

Table 21. Multiple machine security scenarios

WebSphere Application Server security is enabled. LDAP is always deployed.	• Use LDAP as the WebSphere Application Server registry. • Use LDAP as the WebSphere Commerce registry.
	• Use LDAP as the WebSphere Application Server registry. • Use a database as the WebSphere Commerce registry. • You will need to set up LDAP, and place one administrative entry into the LDAP registry.

Table 21. Multiple machine security scenarios (continued)

WebSphere Application Server security is disabled, and your WebSphere Commerce site is located behind a firewall.	• Use a database as the WebSphere Commerce registry. • A WebSphere Application Server registry is not required. • Single sign-on is not supported.
	• Use LDAP as the WebSphere Application Server registry. • A WebSphere Application Server registry is not required.

Note: If you operate your WebSphere Commerce site from behind a firewall, you can disable WebSphere Application Server security. You should only disable WebSphere Application Server security if you are sure that no malicious applications are running behind the firewall.

Security configuration for the Dynamic Cache Monitor

If you are using the WebSphere Application Server Dynamic Cache Monitor to monitor and if the application that you are monitoring has security roles defined in its deployment descriptor you need to do the following:

To navigate to the ″Step: Map security roles to users/groups″ panel in the WebSphere Application Server Administration Console, click **Applications —> Install New Application** and complete the steps (non-security related) required. (For more information, see the topics ″Deploying secured applications″ and ″Assigning users and groups to roles″ topic in the WebSphere Application Server Information Center (`http://www.ibm.com/software/webservers/appserv/infocenter.html`.) In the ″Step: Map security roles to users/groups″ panel:

1. Specify the users and groups that are mapped to each of the security roles.
2. Select the check box beside **Role** as required to select all of the roles or to select individual roles. For each role, you can specify if predefined users such as Everyone or All Authenticated users are mapped to the role. To select specific users or groups from the user registry:
   a. Select a role and click **Lookup users** or **Lookup groups**.
   b. On the **Lookup users** or **Lookup groups** panel that displays, enter the search criteria to extract a list of users or groups from the user registry.
   c. Select individual users or groups from the results displayed.
   d. Click **OK** to map the selected users or groups to the role selected on the ″Step: Map security roles to users/groups panel″.

Currently, there is one role defined which provides access to all of the cache monitor functionality. This means that this page can be used to specify which users can have access to the Dynamic Cache Monitor.

Administering WebSphere Commerce instances through Configuration Manager

If you have WebSphere Application Server global security enabled, you should perform the following steps in order to properly stop, star, create, or delete WebSphere Commerce or WebSphere Commerce Payments instances using the Configuration Manager:

1. In the *WAS_installdir*/properties directory, update the following files and properties to the following values:
   - sas.client.props

     com.ibm.CORBA.securityEnabled=true

     com.ibm.CORBA.loginSource=properties

     com.ibm.CORBA.LoginUserid=validUser

     com.ibm.CORBA.LoginPassword=validPassword
   - soap.client.props

     com.ibm.SOAP.loginUserid=validUser

     com.ibm.SOAP.loginPassword=validPassword

     com.ibm.SOAP.secrityEnabled=true
2. From the *WAS_installdir*/bin directory run the PropFilePasswordEncoder command (on one line) to encode the password in the sas.client.props and soap.client.props files.

   AIX Linux Solaris

   ```
   PropFilePasswordEncoder.sh WAS_installdir/properties/
      sas.client.props com.ibm.CORBA.LoginPassword

   PropFilePasswordEncoder.sh WAS_installdir/properties/
      soap.client.props com.ibm.SOAP.loginPassword
   ```

   400

   ```
   PropFilePasswordEncoder.sh WAS_userdir/WAS_instance/properties/
      sas.client.props com.ibm.CORBA.LoginPassword

   PropFilePasswordEncoder.sh WAS_userdir/WAS_instance/properties/
      soap.client.props com.ibm.SOAP.loginPassword
   ```

   Windows

   ```
   PropFilePasswordEncoder.bat WAS_installdir\properties\
      sas.client.props com.ibm.CORBA.LoginPassword

   PropFilePasswordEncoder.bat WAS_installdir\properties\
      soap.client.props com.ibm.SOAP.loginPassword
   ```
3. Update the config_client script:

   AIX 400 Linux Solaris Add $CLIENTSOAP $CLIENTSAS to the Java argument list. For example:

   ```
   ${JAVA_EXE?} -classpath $CLASSPATH -DIDIR="$WPMDIR"
   -Djava.security.policy="config.policy" -Djava.version="1.3"
   -Dwas.install.root="$WAS_HOME " -Dwas.repository.root="$CONFIG_ROOT"
   -Dcom.ibm.CORBA.BootstrapHost="$COMPUTERNAME" $CLIENTSOAP $CLIENTSAS
   $PM_ARGS -Xmx128m com.ibm.commerce.config.client.CMClient "$@"
   ```

   Windows Add %CLIENTSOAP% %CLIENTSAS% to the Java argument list. For example:

   ```
   "%JAVA_HOME%\bin\java" %CLIENTSOAP% %CLIENTSAS% %PM_ARGS% "
   -Dwas.install.root=%WAS_HOME%" "-Dwas.repository.root=%CONFIG_ROOT%"
   -Dcom.ibm.CORBA.BootstrapHost=%COMPUTERNAME%
   -Djava.security.policy="config.policy"
   com.ibm.commerce.config.client.CMClient %*
   ```

4. Update the `config_server` script:

   AIX 400 Linux Solaris Add `$CLIENTSOAP $CLIENTSAS` to the Java argument list. For example:

   ```
   ${JAVA_EXE?} -classpath $CLASSPATH -DIDIR="$WPMDIR"
   -Djava.security.policy="config.policy"
   -Dwas.install.root="$WAS_HOME " -Dwas.repository.root="$CONFIG_ROOT"
   -Dws.ext.dirs="$WAS_EXT_DIRS" -Dcom.ibm.CORBA.BootstrapHost="$COMPUTERNAME"
   $CLIENTSOAP $CLIENTSAS $PM_ARGS $MAX_HEAP
   com.ibm.commerce.config.server.CMServerImpl "$@"
   ```

   Windows Add `%CLIENTSOAP% %CLIENTSAS%` to the Java argument list. For example:

   ```
   "%JAVA_HOME%\bin\java.exe" %CLIENTSOAP% %CLIENTSAS% %PM_ARGS%
   "-Dwas.install.root=%WAS_HOME%" "-Dwas.repository.root=%CONFIG_ROOT%"
   "-Dws.ext.dirs=%WAS_EXT_DIRS%" -Dcom.ibm.CORBA.BootstrapHost=%COMPUTERNAME%
   -Djava.security.policy="config.policy"
   com.ibm.commerce.config.server.CMServerImpl %*
   ```

Chapter 17. Enabling SSL for production with IBM HTTP Server

400 This section does not apply to the iSeries platform. For iSeries information, see “Enabling SSL on the IBM HTTP Server (iSeries)” on page 191.

After you create your WebSphere Commerce instance with IBM HTTP Server, Secure Sockets Layer (SSL) is enabled for testing purposes. Before you open your site to shoppers, you must enable SSL for production by following the steps in this chapter.

About security

IBM HTTP Server provides a secure environment for your business transactions by using encryption technology. Encryption is the scrambling of information transactions on the Internet so that they cannot be read until they are unscrambled by the receiver. The sender uses an algorithmic pattern or key to scramble (encrypt) a transaction, and the receiver uses a decryption key. These keys are used by the Secure Sockets Layer (SSL) protocol.

Your Web server uses an authentication process to verify the identity of the person with whom you are conducting business (that is, to make sure they are whom they say they are). This involves obtaining a certificate signed by a trusted third party called a certification authority (CA). For IBM HTTP Server users, the CA may be Equifax® or VeriSign® Inc. Other CAs are available as well.

To create a production key file, complete the following steps:
1. Configure a security key file for production.
2. Request a secure certificate from a certifying authority.
3. Set your production key file as the current key file.
4. Receive the certificate and test the production key file.

These steps are described in detail below.

Notes:
1. If you are already using a production key file signed by a certifying authority, you may be able to skip these steps. Read this chapter to make this determination.
2. As you perform these steps, your browser may display security messages. Review the information in each message carefully and decide how to proceed.

Configuring a security key file for production

To configure a security key file for production, do the following on your Web server machine:
1. Stop the IBM HTTP Server.
2. Change your directory to the `conf` subdirectory under the IBM HTTP Server installation directory on your machine.
3. Create a backup copy of `httpd.conf` by and rename the backup copy of the file to `httpd.conf.backup`.

4. Open `httpd.conf` in a text editor.
5. Ensure that the following lines are uncommented (by removing the "#" at the front of the line) for port 443:
   - ▶ Windows
     a. `LoadModule ibm_ssl_module modules/IBMModuleSSL128.dll`
     b. `Listen 443`
     c. `<VirtualHost` *`host.some_domain`*`.com:443>` (You must also substitute your fully qualified host name in this line.)
     d. `SSLEnable`
     e. `</VirtualHost>`
     f. `Keyfile "`*`HTTPServer_installdir`*`/ssl/keyfile.kdb"`
   - ▶ AIX ▶ Linux ▶ Solaris
     a. `LoadModule ibm_ssl_module libexec/mod_ibm_ssl_128.so`
     b. `AddModule mod_ibm_ssl.c`
     c. `Listen 443`
     d. `<VirtualHost` *`host.some_domain`*`.com:443>` (You must also substitute your fully qualified host name in this line.)
     e. `SSLEnable`
     f. `</VirtualHost>`
     g. `SSLDisable`
     h. `Keyfile "`*`HTTPServer_installdir`*`/ssl/keyfile.kdb"`
     i. `SSLV2Timeout 100`
     j. `SSLV3Timeout 1000`
6. Ensure that the following lines are uncommented (by removing the "#" at the front of the line).
   a. For the WebSphere Commerce administrative tools, you require ports 8000, 8002, and 8004:
      ```
      Listen 8000
      Listen 8002
      Listen 8004
      ```
      If you are using WebSphere Commerce Payments, you require ports 5432 and 5433 as well:
      ```
      Listen 5432
      Listen 5433
      ```
   b. Ensure the virtual host sections for the above ports are also uncommented (by removing the "#" at the front of the lines if they are present). You must substitute your fully qualified host name as appropriate in these sections. For a list of the default path name variables in the following examples, see "Path variables" on page ix.

The following examples were derived from the uncommented virtual host sections on a Windows system `httpd.conf` file; these sections are similar on other operating systems.

```
########## IBM WebSphere Payments (Do not edit this section) #################
Listen 5432
Listen 5433
########## End of IBM WebSphere Payments (Do not edit this section) ##########

...

########## IBM WebSphere Commerce (Do not edit this section) #################
Listen 8000
Listen 8002
Listen 8004
########## End of IBM WebSphere Commerce (Do not edit this section) ##########
```

Figure 7. Example "Listen" sections of httpd.conf file

```
########## End of IBM WebSphere Commerce (Do not edit this section) ##########
## VirtualHost: Allows the daemon to respond to requests for more than one
## server address, if your server machine is configured to accept IP packets
## for multiple addresses. This can be accomplished with the ifconfig
## alias flag, or through kernel patches like VIF.
#
## Any httpd.conf or srm.conf directive may go into a VirtualHost command.
## See also the BindAddress entry.
#
#<VirtualHost host.some_domain.com:443>
```

Figure 8. Example virtual host header section of an httpd.conf file

```
########## IBM WebSphere Payments (Do not edit this section) #################
<VirtualHost host.some_domain.com:5433>
SSLEnable
SSLClientAuth 0
ServerName wordsworth.torolab.ibm.com
DocumentRoot "HTTPServer_installdir\htdocs\en_US"
</VirtualHost>
########## End of IBM WebSphere Payments (Do not edit this section) ##########
```

Figure 9. Example virtual host section of httpd.conf file for Payments

```
########## IBM WebSphere Commerce (Do not edit this section) #################
#Instance name : instance_name
<VirtualHost host.some_domain.com:80>
ServerName host.some_domain.com
DocumentRoot "HTTPServer_installdir/htdocs/en_US"
Alias /wcsdoc   "WC_installdir/web/doc"
Alias /wcsstore  "WAS_installdir\installedApps\host\WC_instance_name.ear/Stores.war"
Alias /wcs   "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war"
</VirtualHost>
```

Figure 10. Example virtual host section of httpd.conf file for WebSphere Commerce port 80. (Unsecured port)

```
<VirtualHost host.some_domain.com:443>
SSLEnable
SSLClientAuth 0
ServerName host.some_domain.com
DocumentRoot "HTTPServer_installdir/htdocs/en_US"
Alias /wcsdoc   "WC_installdir/web/doc"
Alias /wcsstore  "WAS_installdir\installedApps\host\WC_instance_name.ear/Stores.war"
Alias /wcs   "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war"
</VirtualHost>
```

Figure 11. Example virtual host section of httpd.conf file for WebSphere Commerce port 443. (Secured port)

```
<VirtualHost host.some_domain.com:8000>
SSLEnable
SSLClientAuth 0
ServerName host.some_domain.com
DocumentRoot "HTTPServer_installdir/htdocs/en_US"
Alias /wcsdoc   "WC_installdir/web/doc"
Alias /wchelp   "WC_installdir/web/doc/en_US"
Alias /adminconsole  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war/tools/adminconsole/wcsadmincon.html"
Alias /wcsstore  "WAS_installdir\installedApps\host\WC_instance_name.ear/Stores.war"
Alias /accelerator  "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war/tools/common/accelerator.html"
Alias /wcs   "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war"
Alias /wcadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war"
Alias /wcorgadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war"
Alias /orgadminconsole "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war/tools/buyerconsole/wcsbuyercon.html"
</VirtualHost>
```

Figure 12. Example virtual host section of httpd.conf file for WebSphere Commerce port 8000. (WebSphere Commerce Accelerator)

```
<VirtualHost host.some_domain.com:8002>
SSLEnable
SSLClientAuth 0
ServerName host.some_domain.com
DocumentRoot "HTTPServer_installdir/htdocs/en_US"
Alias /wcsdoc   "WC_installdir/web/doc"
Alias /wchelp   "WC_installdir/web/doc/en_US"
Alias /adminconsole  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war/tools/adminconsole/wcsadmincon.html"
Alias /wcsstore  "WAS_installdir\installedApps\host\WC_instance_name.ear/Stores.war"
Alias /accelerator  "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war/tools/common/accelerator.html"
Alias /wcs   "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war"
Alias /wcadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war"
Alias /wcorgadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war"
Alias /orgadminconsole "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war/tools/buyerconsole/wcsbuyercon.html"
</VirtualHost>
```

Figure 13. Example virtual host section of httpd.conf file for WebSphere Commerce port 8002. WebSphere Commerce Administration Console

```
<VirtualHost host.some_domain.com:8004>
SSLEnable
SSLClientAuth 0
ServerName host.some_domain.com
DocumentRoot "HTTPServer_installdir/htdocs/en_US"
Alias /wcsdoc   "WC_installdir/web/doc"
Alias /wchelp   "WC_installdir/web/doc/en_US"
Alias /adminconsole  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war/tools/adminconsole/wcsadmincon.html"
Alias /wcsstore  "WAS_installdir\installedApps\host\WC_instance_name.ear/Stores.war"
Alias /accelerator  "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war/tools/common/accelerator.html"
Alias /wcs   "WAS_installdir\installedApps\host\WC_instance_name.ear/CommerceAccelerator.war"
Alias /wcadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/SiteAdministration.war"
Alias /wcorgadmin  "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war"
Alias /orgadminconsole "WAS_installdir\installedApps\host\WC_instance_name.ear/OrganizationAdministration.war/tools/buyerconsole/wcsbuyercon.html"
</VirtualHost>
########## End of IBM WebSphere Commerce (Do not edit this section) ##########
```

Figure 14. Example virtual host section of httpd.conf file for WebSphere Commerce port 8004. WebSphere Commerce Organization Administration Console

Note: It is recommended that your firewall software blocks external access to the ports you have configured for WebSphere Commerce Tools (port 8000, 8002, and 8004 by default). Consult the documentation for the firewall software you are using at your site for information on how you do this.

7. Save your changes.
8. To ensure that your `httpd.conf` file does not contain syntax errors:

   AIX Linux Solaris Change to the `bin` subdirectory under the IBM HTTP Server installation directory on your machine and run the following command: `./httpd -t`

   Windows Change to the IBM HTTP Server installation directory on your machine and run the following command:

   ```
   apache -t
   ```
9. Start the IBM HTTP Server.

Requesting a secure certificate from a certifying authority

To validate the security key file that you just created in the previous step, you need a certificate from a certifying authority (CA) such as Equifax or VeriSign. The certificate contains the server's public key, the Distinguished Name associated with the server's certificate, and the serial number and expiration date of the certificate.

If you want to use a different CA, contact it directly for information on the procedure to follow.

Equifax users

To request a secure server certificate from Equifax, refer to the following Web address and follow the instructions provided:

```
http://www.equifax.com
```

You should receive the secure server certificate through E-mail from Equifax in 2 to 4 business days.

VeriSign users

To request a secure server certificate from VeriSign, refer to the following URL and follow the instructions provided:

```
http://www.verisign.com
```

AIX Although you are using the procedures for IBM HTTP Server, follow the link for **Internet Connection Secure Server** (ICSS). Follow the instructions provided. When you receive your certificate, create the production key file as described in the previous section, if you have not already done so.

Solaris Even though you are using the procedures for IBM HTTP Server, follow the link for **Internet Connection Secure Server** (ICSS). The subsequent page indicates that the procedures apply to the OS/2® and AIX platforms. These instructions also apply for Solaris software.

Follow the instructions provided. Once you submit your request, your certificate should arrive within three to five working days. When you receive it, create the production key file as described in the previous section, if you have not already done so.

Receiving and setting your production key file as the current key file

After the certificate arrives from the CA, you must make the Web server use your production key file. Perform the following steps:

1. Copy the `*certificatename*.kdb`, `*certificatename*.rdb`, and `*certificatename*.sth` files you received from the certificate authority into the `ssl` subdirectory under the IBM HTTP Server installation path on your machine, where *certificatename* is the certificate name you supplied with your certificate request.
2. Stop IBM HTTP Server.
3. AIX Solaris Export `JAVA_HOME` by running the following commands:

   ```
   DISPLAY=host_name:0.0
   export DISPLAY
   JAVA_HOME=java_home
   export JAVA_HOME
   ```

   where *host_name* is the fully qualified host name of the machine you are currently using and *java_home* is:

   - AIX `/usr/java130`
   - Solaris `/opt/WebSphere/AppServer/java131`
4. Open the Key Management Utility (ikeyman).
5. Open the `*certificatename*.kdb` file, and enter your password when prompted.
6. Select **Personal Certificates**, and click **Receive**.
7. Click **Browse**.
8. Select the folder where you have stored the files you received from the certificate authority. Select the `*certificatename*.txt` file and click **OK**.
9. The **Personal Certificates** list box should now list either VeriSign `*certificatename*` certificate or Equifax `*certificatename*` certificate.
10. Exit the Key Management Utility.
11. Change directory to the `conf` subdirectory under the IBM HTTP Server installation path on your machine.
12. Create a backup copy of `httpd.conf`.
13. Open `httpd.conf` in a text editor.
14. Ensure that the lines listed in step 5 on page 186 are not commented.
15. Search for `Keyfile "*keyfile_path_name*/keyfile.kdb"` directive, and change the path name to point to the file created in the above steps.
16. Restart the IBM HTTP Server.

Testing the production key file

To test the production key, do the following:

1. Go the following URL with your browser:

   ```
   https://host_name
   ```

   Notes:

   a. If you have customized your Web server, you may need to type the name of the Web server's front page after the host name.

   b. Be sure to type `https`, *not* `http`.

   If your key is defined correctly, you will see several messages about your new certificate.

2. On the **New Site Certificate** panel, if you want to accept this certificate, select the **Accept this certificate forever (until it expires)** radio button.
3. From your Web browser, restore your caching and proxy (or socks) server settings to their original states.

You have now enabled SSL on your server.

SSL Consideration for WebSphere Commerce Payments

By default, the communication between WebSphere Commerce and WebSphere Commerce Payments is through SSL. However, if you launch the WebSphere Commerce Payments user interface directly as follows, you are invoking WebSphere Commerce Payments using non-SSL communication:

```
http://host_name:port_number/webapp/PaymentManager
```

where *host_name* is your Payments server machine name, and *port_number* is 5432 (by default).

To ensure that the communication is through SSL, use the following URL:

```
https://host_name:port_number/webapp/PaymentManager
```

where *host_name* is your Payments server machine name, and *port_number* is 5433 (by default).

Enhancing confidentiality

When WebSphere Commerce receives a URL request, the Web controller retrieves the interface name for the requested controller command and uses it to look up the implementation class name from the `CMDREG` table. It also determines if HTTPS (secured) protocol is required for the URL request by checking the `HTTPS` column in the `URLREG` table.

Any command which displays sensitive information should have the `HTTPS` value set to a value of “1” (one) in the `URLREG` table. For example, an `OrderProcessView` view command containing details of a customer order should only be transmitted over HTTPS protocol, and therefore the `OrderProcessView` entry in the `URLREG` table has a value of “1” (one) in the `HTTPS` column.

Enabling SSL on the IBM HTTP Server (iSeries)

400 This section applies to the iSeries platform.

SSL is a security protocol. SSL ensures that data transferred between a client and a server remains private. It allows the client to authenticate the identity of the server and the server to authenticate the identity of the client.

Digital certificates are electronic documents that authenticate the servers and clients involved in secured transactions over the Internet. The issuer of digital certificates is called a certificate authority (CA). The iSeries system can perform the role of CA in an Intranet environment issuing server and client certificates, and run as an authenticated server with server certificates issued either by an iSeries CA or an Internet CA like VeriSign. As a Web server, the IBM HTTP Server for iSeries can also be configured to request client certificates for authentication of SSL-enabled clients.

For detailed information on how to enable SSL on the IBM HTTP Server for iSeries, refer to the iSeries Information Center (`http://publib.boulder.ibm.com/html/as400/infocenter.html`). Once you are at the site, select your operating system version and your language, and then click **Go**. Search for the topic "Securing applications with SSL" for guidance on how to enable SSL.

Using SSL with WebSphere Commerce Payments

If you create the system certificate store after creating your WebSphere Commerce instance, you must grant both the WebSphere Commerce Payments instance and the WebSphere Commerce instance access to the system certificate store. For example, the following commands will grant the WebSphere Commerce Payments instance the required access on a V5R1 system:

```
CHGAUT OBJ('/QIBM/UserData/ICSS/Cert/Server') USER(QPYMSVR) DTAAUT(*RX)
CHGAUT OBJ('/QIBM/UserData/ICSS/Cert/Server/DEFAULT.KDB') USER(QPYMSVR) DTAAUT(*R)
```

and the following commands will grant the WebSphere Commerce the required access on a V5R1 system:

```
CHGAUT OBJ('/QIBM/UserData/ICSS/Cert/Server') USER(QEJBSVR) DTAAUT(*RX)
CHGAUT OBJ('/QIBM/UserData/ICSS/Cert/Server/DEFAULT.KDB') USER(QEJBSVR) DTAAUT(*R)
```

If you choose to use a remote WebSphere Commerce Payments instance, you must configure both the WebSphere Commerce instance and the WebSphere Commerce Payments instance to trust the remote certificate authority that issues the digital certificate. To establish a trust relationship between the two remote applications, refer to the following high-level procedure:

1. On the WebSphere Commerce machine, use the Digital Certificate Manager to export the server's certificate authority.
2. Transfer the certificate file to the WebSphere Commerce Payments machine.
3. On the WebSphere Commerce Payments machine, use the Digital Certificate Manager to import the WebSphere Commerce server's certificate authority.
4. Configure the WebSphere Commerce Payments application server to trust the imported WebSphere Commerce server's certificate authority.
5. On the WebSphere Commerce Payments machine, use the Digital Certificate Manager to export the server's certificate authority.
6. Transfer the certificate file to the WebSphere Commerce machine.
7. On the WebSphere Commerce machine, use the Digital Certificate Manager to import the WebSphere Commerce Payments server's certificate authority.
8. Configure the WebSphere Commerce application server to trust the imported WebSphere Commerce Payments server's certificate authority.

For detailed information refer to the following Web address, and look for **Hints and Tips**: WebSphere Commerce Technical Library Web page (`http://www.software.ibm.com/software/commerce/wscom/library/lit-tech.html`)

+ # Enabling the SSL Accelerator option

+ When using an SSL Accelerator (also known as an SSL Terminator) with
+ WebSphere Commerce, you can use the SSL Accelerator option
+ (SSLAcceleratorOption) to configure WebSphere Commerce to correctly receive
+ requests that require redirects. An SSL Accelerator (or SSL Terminator) is hardware
+ that strips off HTTPS (secure sockets layer) encryption at or before the Web server
+ tier in a multi-tier setup.

To enable the SSL Accelerator option, the following attributes have to be added to the *WC_instance*.xml file in the WebServer node:

```
<WebServer ...
:
SSLAcceleratorOption="Enabled"
InSSLPort="443"
InNonSSLPort="80"
OutSSLPort="443"
OutNonSSLPort="80"
:
... >
```

Where:

InSSLPort
: The port configured for WebSphere Commerce to receive the SSL data - WebSphere Commerce will treat this as SSL data even if the scheme says http. The default is port 443.

InNonSSLPort
: The port configured for WebSphere Commerce to receive the non SSL data - WebSphere Commerce will treat any data received in this port as non-SSL data. The default port is 80.

OutSSLPort
: The port that WebSphere Commerce will use to send out SSL data on a redirect. The default port is 443.

OutNonSSLPort
: The port that WebSphere Commerce will use to send out non-SSL data on a redirect. The default port is 80.

Chapter 18. Enabling SSL for IBM Directory Server (LDAP)

The following are the steps to configure SSL security for IBM Directory Server and WebSphere Commerce.

Setting up IBM Directory Server

▶ 400 This section does not apply to the iSeries platform. For iSeries information, see "Setting up IBM OS/400 Directory Services on the iSeries platform."

To set up the IBM Directory Server:

1. Install IBM Directory Server according to the IBM Directory Server product installation instructions. Ensure that you install the GSKit component.
2. After the installation completes, invoke the IBM Key Manager by running the gsk5ikm executable.
3. Create a new CMS Key database file. Make sure the **stash password to file** is selected (for example, `ldap_key.kdb`)
4. Create a self-signed certificate using X509 V3 version and 1024 key size. (You can assign a meaningful label to the certificate, for example, your name.)
5. Extract the certificate as a certificate file (for example, `cert.arm`) using the `Base64-encoded ASCII data` data type.
6. Open a browser to the following address: `http://`*`host_name`*`/ldap` where *host_name* is your LDAP server machine name.
7. Click **Security —> SSL —> Settings** and make the following changes:
   - SSL status: SSL on or SSL only
   - Authentication method: Server Authentication
   - Secure port: 636
   - Key database path and file name:

     ▶ AIX ▶ Linux ▶ Solaris `/Keys/ldap_key.kdb`

     ▶ Windows *`drive`*`:\Keys\ldap_key.kdb`
   - Key label: *your_label* (The label of the certificate.)
   - Key password: *xxxxx* (The password of the CMS Key database file. If you choose '**stash the password to a file**', you do not need to input the password.)
8. Click **Update** and restart SecureWay.

Setting up IBM OS/400 Directory Services on the iSeries platform

▶ 400 To set up the IBM OS/400 Directory Services on iSeries:

1. Install IBM iSeries Access for Windows.
2. Start the iSeries Navigator on a Windows machine by selecting **Start** —> **Programs** —> **IBM iSeries Access for Windows** —> **iSeries Navigator**.
3. Create a connection to the target iSeries machine if no connection for the machine exists.
4. Expand the target machine in the left panel, then expand **Network** —>**Servers** in the left panel.

5. Click **TCP/IP** in the left panel.
6. Right-click **Directory** in the right panel and select **Properties** from the popup menu.
7. In the Directory Properties window, click the **Network** tab.
8. Click **Digital Certificate Manager** to launch Digital Certificate Manager and assign a certificate to Application "`Directory Services server`".
9. After assigning the certificate to the Directory Services server, click **OK** to close the Directory Properties window
10. Re-open the Directory Properties window, and you will see that Secure Socket Layer (SSL) is enabled. You can accept the default settings:
    - SSL status:
    - Authentication Method: Server Authentication
    - Secure port: 636
11. Restart the Directory Services server.

Assigning and importing a self-signed certificate to WebSphere Application Server

400 If your SSL Certificate has not been issued by a Certificate Authority (CA), such as VeriSign or Thwate, you should export the local CA from an iSeries machine and import it to the default trust keystore on the WebSphere Commerce machine. To enable SSL with iSeries local certificate and export the local CA from an iSeries machine, do the following:

1. Make sure the `HTTP *Admin` server is up. If it is not, run:
   ```
   STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN)
   ```
2. Open the iSeries Task page by launching a browser to the following address: `http://host name:2001/`.
3. Select **Digital Certificate Manager**.
4. Click on **Select a Certificate Store**.
5. From the Certificate Store, select ***System**.
6. If you do not see the **Install Local CA Certificate on Your PC** link, then you need to create a local CA:
   a. a. Click Create a Certificate Authority (CA).
   b. Restart the `*Admin` HTTP Server on iSeries.
   c. Create the new certificate as a Client or Server type.
   d. Select the newly created Local Certificate Authority.
   e. Assign this certificate to Directory Services server.
7. Click **Install Local CA Certificate** on your PC.
8. Click Install Certificate. Then save the certificate (`.cer` file) in a temporary folder.
9. Import the certificate authority (`.cer` file) to Microsoft Internet Explorer and then export the certificate authority again to a `.cer` file (Binary 64 Encoding) in a temporary directory.
10. Import the certificate (Binary 64 Encoding) into the WebSphere Application Server trust keystore. For example:
    ```
    keytool -import -alias nck -file /temp_dir/nck.cer
           -keystore /qibm/proddata/java400/jdk13/lib/security/cacerts
    ```

WebSphere Application Server

In WebSphere Application Server:

1. Launch the IKeyMan (IBM Key Manager) that is provided with WebSphere Application Server. (You can find it from the WebSphere Application Server menu or you can directly type `ikeyman` in a command window.)

   Note: This IBM Key Manager is different from that provided by SecureWay The default password is '`changeit`'.
2. Open the WebSphere Application Server carcerts keystore (for example, *`WAS_installdir`*`\AppServer\java\jre\lib\security\cacerts` on Windows)
3. Traverse to **Signer Certificates**, then click **Add**. Use '**Base64-encoded ASCII data**' data type, and choose the certificate file you created in step 5 on page 195.
4. Enter a name for the certificate.
5. Close IKeyMan.

WebSphere Commerce

To set up WebSphere Commerce to work with SecureWay Directory Server you need to modify the *`instance`*`.xml` file:

1. Add a new JNDI environment variable:
   ```
   java.naming.security.protocol = ssl
   ```
2. Change `LdapPort` to '636':
   ```
   LdapPort = 636
   ```
3. Restart WebSphere Commerce.

The following is a example:

```
<MemberSubSystem  name="Member SubSystem"
          AuthenticationMode="LDAP"
          ProfileDataStorage="LDAP"  >

     <Directory LdapAdminDN="cn=root"
                 LdapAuthenticationMode="SIMPLE"
                 LdapTimeOut="0"
                 LdapVersion="3"
                 EntryFileName="E:/WebSphere/WPS/xml/ldap/attributeMap.xml"
                 LdapPort="636"
                 LdapAdminPW="<adminpassword>"
                 LdapHost="<hostname>"
                 MigrateUsersFromWCSdb="OFF"
                 JNDIEnvPropName1="java.naming.security.protocol"
                 JNDIEnvPropValue1="ssl"
                 display="false"
                 LdapType="SECUREWAY"

   . . . .

  />


</MemberSubSystem>
```

Part 6. Appendixes

Appendix. Default access control policies and groups

The Appendix lists the default policies and groups provided with WebSphere Commerce.

Default access control policies

The default access control policies are organized into the following categories:

- **Role-based policies:** The role-based policies for each default role. These policies are also referred to as command-level policies because they define who can execute each command.
- **Resource-level policies:** The resource-level policies, grouped by business area. These policies define the actions a group of users can perform on specific resources. Under each business area, policies are organized by the type of resource they regulate:
  - **Data resources** - business objects that can be manipulated such as an order or a bid.
  - **DataBean resources** - contain information about business objects. DataBeans are used to display object information on a Web page.

Table 22. Where to find information on policies

Policies	**Starting on page**
Role-based policies	“Role-Based Policies” on page 202
Resource-level policies by business area	“Resource-Level Policies By Business Area” on page 205
Orders	“Orders” on page 205
Trading (contracts)	“Trading (Contracts)” on page 206
Approvals	“Approvals” on page 206
Auctions	“Auctions” on page 207
Business Intelligence	“Business Intelligence” on page 207
Membership	“Membership” on page 207
Marketing	“Marketing” on page 208
Catalog	“Catalog” on page 209
Connectivity and notification	“Connectivity and Notification” on page 209
Procurement	“Procurement” on page 209
Coupons	“Coupons” on page 209
Customer profiling	“Customer Profiling” on page 210
Discounts	“Discounts” on page 210
Scheduled Inventory	
Inventory management	
Order management	“Order Management” on page 211
Payments	“Payments” on page 211
Policy Editor	“Policy Editor” on page 211
Product Advisor	“Product Advisor” on page 211
RFQ	“RFQ” on page 212
Rules	“Rules” on page 212
Scheduler	“Scheduler” on page 212
Commerce Accelerator	“Commerce Accelerator” on page 212

Table 22. Where to find information on policies (continued)

Policies	Starting on page
Shipping	"Shipping" on page 213
Taxation	"Taxation" on page 213
Live Help/Collaborative Workspaces/Customer care	"Live Help/ Collaborative Workspaces/Customer Care" on page 213
Store state	"Store state" on page 213
Store Management	

Role-Based Policies

- SiteAdministratorsCanDoEverything
- BuyerAdministratorsExecuteBuyersAdministratorsCommands
- BuyerApproversExecuteBuyerApproversCmdResourceGroup
- GuestsExecuteGuestUsersCmdResourceGroup
- BecomeUserCustomerServiceGroupExecutesBecomeUserCmdsResourceGroup
- CustomerServiceRepresentativesExecuteCustomerServiceRepCmdResourceGroup
- MarketingManagersExecuteMarketingManagerCmdResourceGroup
- CustomerServiceSupervisorsExecuteCustomerServiceSupervisorCmdResourceGroup
- AccountRepresentativesExecuteAccountRepresentativesCmdResourceGroup
- SalesManagersExecuteSalesManagersCmdResourceGroup
- ProductManagersExecuteProductManagersCmdResourceGroup
- SellerAdministratorsExecuteSellerAdministratorsCommands
- SellersExecuteSellersCmdResourceGroup
- CategoryManagersExecuteCategoryManagersCmdResourceGroup
- Buyers(buy-side)ExecuteBuyers(buy-side)CommandsResourceGroup
- Buyers(sell-side)ExecuteBuyers(sell-side)CommandsResourceGroup
- PickPackersExecutePickPackersCmdResourceGroup
- ReceiversExecuteReceiversCmdResourceGroup
- ReturnsAdministratorsExecuteReturnsAdministratorsCmdResourceGroup
- OperationsManagersExecuteOperationsManagersCmdResourceGroup
- LogisticsManagersExecuteLogisticsManagersCmdResourceGroup
- ProcurementBuyersExecuteProcurementBuyersCmdResourceGroup
- CustomerServiceRepresentativesExecuteCustomerServiceRepresentativeViews
- BuyerAdministratorsExecuteBuyerAdministratorsViews
- BuyerApproversExecuteBuyerApproversViews
- MarketingManagersExecuteMarketingManagersViews
- CustomerServiceSupervisorsExecuteCustomerServiceSupervisorViews
- SalesManagersExecuteSalesManagersViews
- AccountRepresentativesExecuteAccountRepresentativesViews
- Buyers(buy-side)ExecuteBuyers(buy-side)Views
- Buyers(sell-side)ExecuteBuyers(sell-side)Views
- CategoryManagersExecuteCategoryManagersViews
- CustomersExecuteCustomersViews
- ProductManagersExecuteProductManagersViews

- PickPackersExecutePickPackersViews
- ReceiversExecuteReceiversViews
- ReturnsAdministratorsExecuteReturnsAdministratorsViews
- OperationsManagersExecuteOperationsManagersViews
- LogisticsManagersExecuteLogisticsManagersViews
- SellerAdministratorsExecuteSellerAdministratorsViews
- SellersExecuteSellersViews
- RegisteredApprovedUsersExecuteRegisteredApprovedUsersViews
- NonRejectedUsersExecuteNonRejectedUsersViews
- GuestUsersExecuteGuestUsersViews
- RegisteredApprovedUsersExecuteRegisteredApprovedUsersCommandsResourceGroup
- ChannelManagersExecuteChannelManagersCommands
- AllUsersExecuteAllSiteUserCmdResourceGroup
- AllUsersExecuteAllSiteUsersViews
- RegisteredCustomersForOrgExecuteRegisteredUserCmdResourceGroup
- RegisteredCustomersForOrgExecuteRegisteredUserViews
- ChannelManagersExecuteChannelManagersViews
- AllUsersExecuteResellerUserCmdResourceGroup
- AllUsersExecuteResellerUserViews
- RegisteredCustomersForOrgExecuteRegisteredResellerUserCmdResourceGroup
- RegisteredCustomersForOrgExecuteRegisteredResellerUserViews

The following table displays the role-based policies by role, access group, resource group, and view.

Notes:

1. Most items in the table except for the **Role** column have been split across each cell for display purposes as they are lengthy.
2. Not all of the roles below are defined roles in WebSphere Commerce. For more information on defined WebSphere Commerce roles, see "Roles" on page 28.

Table 23. Role-based policies by role, access group, resource group, and view

Role	Access Group used in role-based policies	Resource Group used in role-based policies for Controller commands	Action Group used in role-based policies for Views
Site Administrator	`SiteAdministrators`	`n/a`	`n/a`
Buyer Administrator	`BuyerAdministrators`	`BuyerAdministrators CommandsResource Group`	`BuyerAdministrators Views`
Buyer Approver	`BuyerApprovers`	`BuyerApproversCmd ResourceGroup`	`BuyerApproversViews`
Guest[1]	`Guests`	`GuestUsersCmd ResourceGroup`	`GuestUsersViews`
Customer Service Representative	`CustomerService Representatives`	`CustomerService RepCmdResourceGroup`	`CustomerService Representative Views`
Marketing Manager	`MarketingManagers`	`MarketingManager CmdResourceGroup`	`MarketingManagersViews`
Customer Service Supervisor	`CustomerService Supervisors`	`CustomerService Supervisor CmdResourceGroup`	`CustomerService SupervisorViews`

Table 23. Role-based policies by role, access group, resource group, and view (continued)

Role	Access Group used in role-based policies	Resource Group used in role-based policies for Controller commands	Action Group used in role-based policies for Views
Account Representative	AccountRepresentatives	AccountRepresentativesCmdResourceGroup	AccountRepresentativesViews
Sales Manager	SalesManagers	SalesManagersCmdResourceGroup	SalesManagersViews
Product Manager	ProductManagers	ProductManagersCmdResourceGroup	ProductManagersViews
Seller Administrator	SellerAdministrators	SellerAdministratorsCommandsResourceGroup	SellerAdministratorsViews
Seller	Sellers	SellersCmdResourceGroup	SellersViews
Category Manager	CategoryManagers	CategoryManagersCmdResourceGroup	CategoryManagersViews
Buyer (buy-side)	Buyers(buy-side)	Buyers(buy-side)CommandsResourceGroup	Buyers(buy-side)Views
Buyer (sell-side)	Buyers(sell-side)	Buyers(sell-side)CommandsResourceGroup	Buyers(sell-side)Views
Pick Packer	PickPackers	PickPackersCmdResourceGroup	PickPackersViews
Receiver	Receivers	ReceiversCmdResourceGroup	ReceiversViews
Returns Administrator	ReturnsAdministrators	ReturnsAdministratorsCmdResourceGroup	ReturnsAdministratorsViews
Operations Manager	OperationsManagers	OperationsManagersCmdResourceGroup	OperationsManagersViews
Logistics Manager	LogisticsManagers	LogisticsManagersCmdResourceGroup	LogisticsManagersViews
Procurement Buyer	ProcurementBuyers	ProcurementBuyersCmdResourceGroup	n/a
Registered Approved User[2]	RegisteredApprovedUsers	RegisteredApprovedUsersCommandsResourceGroup	RegisteredApprovedUsersViews
Non-Rejected User[3]	NonRejectedUsers	NonRejectedUserCommandsResourceGroup	NonRejectedUsersViews
Channel Manager	ChannelManagers	ChannelManagersCmdResourceGroup	ChannelManagersViews
All Users[4]	AllUsers	ResellerUserCmdResourceGroup[5]	ResellerUserViews[5]
		AllSiteUserCmdResourceGroup[6]	AllSiteUsersViews[6]
Registered Customer (with OrgandAncestorOrgs role qualifier)	RegisteredCustomersForOrg	RegisteredUserCmdResourceGroup	RegisteredUserViews
		RegisteredResellerUserCmdResourceGroup	RegisteredResellerUserViews

Table 23. Role-based policies by role, access group, resource group, and view (continued)

Role	Access Group used in role-based policies	Resource Group used in role-based policies for Controller commands	Action Group used in role-based policies for Views

Notes:

1. "Guest" is not a true role. Users who have a registration status set to "G" (the `USER.REGISTERTYPE` column is set to "G") implicitly belong to the `Guests` access group.
2. "Registered Approved User" is not a true role. Users who have a registration status set to "R" (the `USER.REGISTERTYPE` column column is set to "R") and whose status is approved (the `MEMBER.STATE` column is set to 1) implicitly belong to the `RegisteredApprovedUsers` access group.
3. "Non-Rejected User" is not a true role. Users whose registration status is not-rejected (`MEMBER.STATE` column is not set to 2) implicitly belong to the `NonRejectedUsers` access group.
4. "All Users" is not a true role. All users in the system implicitly belong to the `AllUsers` access group.
5. These action groups and resource groups belong to policies that are part of the B2CPolicyGroup. This policy group likely applies only to organizations that follow the B2C business model.
6. These action groups and resource groups belong to policies that are part of the ManagementAndAdministrationPolicyGroup. This policy group likely applies to all organizations.

Resource-Level Policies By Business Area

Orders

Data resources: order:

- AllUsersExecuteAllUsersActionGroupCommandsOnOrderResource
- AllUsersExecuteOrderCreateCommandsOnStoreResource
- AllUsersExecuteOrderReadCommandsOnOrderResource
- AllUsersExecuteOrderPrepareCommandsOnOrderResource
- AllUsersExecuteOrderWriteCommandsOnOrderResource
- AllUsersExecuteScheduledOrderCancelOnOrderResource
- AllUsersExecuteReturnAgainstOrderOnOrderResource
- AllUsersExecuteOrderProcessOnOrderResource
- OrderManagersForOrgExecuteOrderManageCommandsOnOrderResource
- CustomerOrderManagersForOrgExecuteOrderProcessOnOrderResource
- ResellerAdministratorsForOrgExecuteOrderReadCommandsOnOrderDataResourceGroup
- ResellerAdministratorsForOrgExecuteOrderPrepareCommandsOnOrderDataResourceGroup
- ResellerAdministratorsForOrgExecuteOrderWriteCommandsOnOrderDataResourceGroup
- ResellerAdministratorsForOrgExecuteScheduledOrderCancelOnOrderDataResourceGroup
- ResellerAdministratorsForOrgExecuteOrderProcessOnOrderDataResourceGroup
- EmailOrderNotificationManagersForOrgExecuteCustomerServiceEmailOrderOnOrderResource

Data resources: requisition list:

- AllUsersExecuteRequisitionListCreateCommandsOnStoreEntityResource
- AllUsersExecuteRequisitionListSharedReadCommandsOnSharedRequisitionListResource
- AllUsersExecuteRequisitionListExclusiveReadCommandsOnPrivateRequisitionListResource
- AllUsersExecuteRequisitionListWriteCommandsOnRequisitionListResource
- AllUsersExecuteRequisitionListSharedProcessCommandsOnSharedRequisitionListResource
- AllUsersExecuteRequisitionListExclusiveProcessCommandsOnPrivateRequisitionListResource

Data resources: interest item:

- AllUsersExecuteInterestItemReadCommandsOnInterestItemListResource
- AllUsersExecuteInterestItemWriteCommandsOnInterestItemListResource

Data resources: RMA:
- AllUsersExecuteRMACreateCommandsOnStoreResource
- AllUsersExecuteRMAReadCommandsOnRMAResource
- AllUsersExecuteRMAPrepareOnRMAResource
- AllUsersExecuteRMAWriteCommandsOnRMAResource
- AllUsersExecuteRMAProcessCommandsOnRMAResource
- RMAApproversForOrgExecuteRMAApproveCommandsOnRMAResource
- RMADisposersForOrgExecuteRMADisposeCommandsOnRMAResource
- RMAReceiversForOrgExecuteRMAReceiveCommandsOnRMAResource
- RMAManagersForOrgExecuteRMAManageCommandsOnRMAResource
- StoreAdministratorsForOrgExecuteRMACreditCommandsOnStoreEntityResource

Databeans: order:
- AllUsersDisplayOrderDatabeanResourceGroup
- AllUsersDisplayApprovalsOrderDataBeansResourceGroup
- AccountRepresentativesForOrgDisplayOrderDatabeanOnlyResourceGroup

Databeans: requisition list:
AllUsersDisplaySharedRequisitionListDataBeansIfSameOrganizationalEntityAsCreator

Databeans: interest item: AllUsersDisplayInterestItemDatabeanResourceGroup

Databeans: RMA: AllUsersDisplayRMADatabeanResourceGroup

Trading (Contracts)

Data resources: contract:
- ContractCreatorsForOrgExecuteContractCreateCommandsOnMemberResource
- ContractManagersForOrgExecuteContractManageCommandsOnContractResource
- ContractOperatorsForOrgExecuteContractDeployCommandsOnContractResource
- ContractViewersExecuteContractDisplayCommandsOnContractResource
- ContractOperatorsForOrgExecuteContractSubmitCommandsOnContractResource
- ContractManagersForOrgExecuteContractAccountManageCommandsOnAccountResource

Data resources: business policy:

- BusinessPolicyAdministratorsForOrgExecuteBusinessPolicyCreateCommandsOnStoreResource
- BusinessPolicyAdministratorsForOrgExecuteBusinessPolicyManageCommandsOnBusinessPolicyResource

Data resources: store creation:
StoreCreatorsForOrgExecuteStoreCreationCommandsOnOrganizationResource

Databeans: AccountHandlersForOrgDisplayTradingDatabeanResourceGroup

Approvals

Data resources:
- AllUsersExecuteApproveCommandsOnApprovalResource
- FlowAdministratorExecutesFlowAdminCreateCommandsOnStoreEntityResource

- FlowAdministratorExecutesFlowadminDeleteCommandsOnFlowadminResource

Databeans: FlowAdministratorsForOrgDisplayFlowadminDatabean

Auctions

Data resources:

- AuctionAdministratorsForOrgExecuteAuctionCreateCommandsOnStoreEntityResource
- AuctionAdministratorsForOrgExecuteAuctionManageCommandsOnAuctionResource
- AuctionManagersForOrgExecuteAdminRetractBidCommandsOnAuctionResource
- AuctionAdministratorsForOrgExecuteAuctionStyleCreateCommandsOnStoreEntityResource
- AuctionAdministratorsForOrgExecuteAuctionStyleManageCommandsOnAuctionStyleResource
- AuctionAdministratorsForOrgExecuteBidControlRuleCreateCommandsOnStoreEntityResource
- AuctionAdministratorsForOrgExecuteBidControlRuleManageCommandsOnBidControlRuleResource
- RegisteredApprovedUsersExecuteBidCreateCommandsOnAuctionResource
- RegisteredApprovedUsersExecuteBidManageCommandsOnBidResources
- RegisteredApprovedUsersExecuteAutoBidCreateCommandsOnAuctionResource
- RegisteredApprovedUsersExecuteAutoBidManageCommandsOnAutoBidResources

Databeans: AuctionDatabeanOwnersDisplayAuctionDatabeans

Business Intelligence

Data resources:

- BusinessAnalystsForOrgExecuteViewContextListCommandsOnStoreEntityResource
- IntelligenceReportViewersForOrgExecuteViewBusinessIntelligenceReportCommands OnStoreEntityResource

Membership

Data resources: user:

- MembershipAdministratorsForOrgExecuteUserAdminUpdateCommandsOnUserResource
- GuestsExecuteUserSelfRegistrationCommandsOnOrganizationResource
- NonRejectedUsersExecuteUserSelfRegistrationContinuationCommandsOnUserResource
- NonRejectedUsersExecuteNonRejectedUserCommands
- AllUsersDisplayUserDatabeanResourceGroup
- NonRejectedDisplayUserDatabeanResourceGroup

Data resources: organization:

- OrgAdminConsoleMembershipAdministratorsForOrgExecuteOrgEntityPolicySubscription UpdateCommandsOnOrganizationResource
- OrgAdminConsoleMembershipAdministratorsForOrgExecuteOrganizationManageActions OnOrganiztionResource
- CSAMembershipAdministratorsForOrgExecuteUserAdminRegistrationCommands OnOrganizationResource
- CSAMembershipAdministratorsExecuteUserAdminRegistrationCommands OnOrganizationResource
- MembershipAdministratorsForOrgExecuteOrgEntityRegistrationCommands OnOrganizationResource
- MembershipAdministratorsForOrgExecuteOrgEntityUpdateCommandsOnOrganizationResource GuestsExecuteResellerSelfRegistrationCommandsOnOrganizationResource
- NonRejectedUsersExecuteResellerSelfRegistrationContinuationCommandsOnOrganizationResource

- ChannelManagersExecuteOrgEntityLockCommandsOnOrgResource
- OrgAdminConsoleMembershipAdministratorsForOrgExecuteApproveGroupUpdateCommands OnOrganizationResource

Data resources: member group:

- OrgAdminConsoleMembershipAdministratorsForOrgExecuteMemberGroupMemberUpdate CommandsOnUserResource
- OrgAdminConsoleMembershipAdministratorsForOrgExecuteMemberGroupMemberUpdate CommandsOnMemberGroupResource
- MemberGroupAdministratorsForOrgExecuteMemberGroupCreateCommandsOnMemberResource
- MemberGroupManagersForOrgExecuteMemberGroupManageCommandsOnMemberGroupResource

Data resources: address:

- NonRejectedUsersExecuteAddressManageCommandsOnUserResource
- MembershipAdministratorsForOrgExecuteAddressManageCommandsOnMemberResource

Data resources: role:

- MembershipAdministratorsForOrgExecuteRoleUnassignCommandsOnUserResource
- OrganizationRoleAdministratorsExecuteRoleManageCommandsOnOrganizationResource
- MembershipAdministratorsForOrgExecuteUserRoleAssignCommandsOnOrganizationResource

Data resources: member attribute:

- OrgAdminConsoleMembershipAdministratorsForOrgExecuteMemberAttributeCommands OnOrgResource
- AllUsersExecuteMemberAttributeCommandsOnUserResource

Databeans:

- MembershipViewersForOrgDisplayMembershipDatabeanResourceGroup
- MembershipAdministratorsForOrgDisplayOrganizationDatabeanResourceGroup
- MembershipAdministratorsForOrgDisplayUserDatabeanResourceGroup
- EmployeesDisplayOrganizationSpecificDatabeanResourceGroup

Marketing

Data resources: campaigns:

- CampaignManagersForOrgExecuteCampaignRelatedCreateCommandsOnStoreEntityResource
- CampaignManagersForOrgExecuteCampaignUpdateCommandsOnCampaignResource
- CampaignManagersForOrgExecuteInitiativeUpdateCommandsOnInitiativeResource
- CampaignManagersForOrgExecuteEMarketingSpotUpdateCommandsOnEMarketingSpotResource
- CampaignManagersForOrgExecuteCollateralUpdateCommandsOnCollateralResource

Data resources: e-mail activities:

- EmailActivityEditorsForOrgExecuteEmailActivitySaveCommandsOnEmailActivity DataResourceGroup
- EmailActivityEditorsForOrgExecuteEmailActivitySaveCommandsOnStoreEntity DataResourceGroup
- EmailActivityEditorsForOrgExecuteEmailActivityDeleteCommandsOnEmailActivity DataResourceGroup
- EmailActivityConfigurationEditorsForOrgExecuteEmailActivityConfigurationSaveCommands OnEmailActivityDataResourceGroup
- EmailActivityConfigurationEditorsForOrgExecuteEmailActivitySaveCommandsOnStoreEntity DataResourceGroupAllUsersExecuteEmailOptOutDataResourceGroup

Databeans: campaigns: CampaignManagersForOrgDisplayCampaignDatabeanResourceGroup

Databeans: e-mail activities:
- EmailUserReceiveDataBeanPolicy
- EmailActivityDataBeanPolicy
- EmailConfigurationDataBeanPolicy

Databeans: e-promotions: EpromotionDisplayDataBeanPolicy

Catalog

Data resources:
- CatalogManagersForOrgExecuteStoreCategoryManageCommandsOnCatalogResource
- CatalogManagersForOrgExecuteCatalogManageCommandsOnCatalogResource
- CatalogGroupManagersForOrgExecuteCatalogGroupManageCommandsOnCatalogGroupResource
- CatalogEntryManagersForOrgExecuteStoreCatalogEntryManageCommandsOnStoreEntityResource
- CatalogGroupManagersForOrgExecuteProductSetAddCommandsOnCatalogResource
- CatalogGroupManagersForOrgExecuteProductSetManageCommandsOnProductSetResource
- CatalogEntryManagersForOrgExecuteCatalogEntryManageCommandsOnCatalogEntryResource
- CatalogEntryManagersForOrgExecuteCatalogEntryRelationManageCommandsOnCatalogResource
- CatalogEntryManagersForOrgExecuteCatalogStoreManageCommandsOnStoreEntityResource

Databeans:
- ProductAdministratorsForOrgDisplayProductDataBeansResourceGroup
- CatalogGroupViewersForOrgDisplayCatalogGroupDataBeansResourceGroup
- CatalogListViewersForOrgDisplayCatalogListDataBeansResourceGroup

Connectivity and Notification

Data resources:

- BackendOrderAdministratorsForOrgExecuteBackendOrderStatusCreateCommandsOnOrderDataResource
- BackendPickPackersForOrgExecuteBackendPickPackListCommandsOnFulfillmentCenterDataResource
- MessagingUpdateAdministratorsForOrgExecuteMessagingUpdateCommandsOnStoreEntityResource

Procurement

Data resources:

- ProcurementAdministratorsForOrgExecuteProcurementAuthenticationAndRegistration OnOrganizationResource
- ProcurementShoppingCartManagersExecuteProcurementShoppingCartManageOnOrderResource

Coupons

Data resources:
- CouponAdministratorsForOrgExecuteCouponPromotionCreateCommandsOnStoreEntityResource
- CouponAdministratorsForOrgExecuteCouponPromotionDeleteCommandsOnCouponPromotionResource
- AllUsersExecuteCouponRedemptionCommandsOnCouponWalletResource
- AllUsersExecuteCouponDeleteCommandsOnCouponWalletResource
- CouponAdministratorsForOrgExecuteCouponPromotionUpdateCommandsOnStoreEntityResource
- AllUsersExecuteCouponSaveCommandsOnCouponWalletResource

Databeans: CouponAdministratorsForOrgDisplayECouponPromotionBeans

Customer Profiling

Data resources: CustomerProfileEditorsForOrgExecuteSegmentManageCommandsOnStoreEntityResource

Databeans: CustomerProfileEditorsForOrgDisplaySegmentationDatabeansResourceGroup

Discounts

Data resources:
- DiscountAdministratorsForOrgExecuteDiscountCreateCommandsOnStoreEntityResource
- DiscountAdministratorsForOrgExecuteDiscountDeployCommandsOnCalculationCodeResource
- DiscountAdministratorsForOrgExecuteDiscountAssociateCommandsOnCalculationCodeResource

Databeans: DiscountViewersForOrgDisplayDiscountDatabeans

Inventory Management

Data resources:
- FulfillmentCenterManagersForOrgExecuteFulfillmentCenterCreateCommandsOn OrganizationResource
- FulfillmentCenterManagersForOrgExecuteFulfillmentCenterManageCommandsOn FulfillmentCenterResource
- PickBatchInventoryManagersForOrgExecuteReleaseReadyShipCommandsOn FulfillmentCenterResource
- VendorInventoryManagersForOrgExecuteVendorManageCommandsOnVendorResource
- VendorInventoryManagersForOrgExecuteVendorCreateCommandsOnStoreEntityResource
- ExpectedInventoryManagersForOrgExecuteInventoryManageCommandsOnStoreEntityResource
- PickPackGeneratorsForOrgExecutePickPackGenerateCommandsOnFulfillmentCenterResource
- InventoryAdjustersForOrgExecuteInventoryAdjustCommandsOnStoreEntityResource
- ReturnReasonsManagersForOrgExecuteReturnReasonsCommandsOnStoreEntityResource
- FulfillmentCenterManagersForOrgExecuteFulfillmentCenterReleaseOnFulfillmentCenter ReleaseDataResourceGroup
- SharedFulfillmentCenterPickBatchInventoryManagersExecuteReleaseReadyShipCommands OnFulfillmentCenterDataResource
- SharedFulfillmentCenterPickPackGeneratorsExecutePickPackGenerateCommands OnFulfillmentCenterResource
- SharedFulfillmentCenterManagersExecuteFulfillmentCenterReleaseCommandsOnFulfillment CenterReleaseDataResourceGroup

Databeans:
- ReturnReasonsManagersForOrgDisplayReturnReasonsOrderManagementDataBeansResourceGroup
- ExpectedInventoryManagersForOrgDisplayExpectedInventoryDataBeansResourceGroup
- VendorInventoryManagersForOrgDisplayVendorInventoryDataBeansResourceGroup
- ProductFindInventoryManagersForOrgDisplayProductFindInventoryDataBeansResourceGroup
- FulfillmentCenterManagersForOrgDisplayFulfillmentCenterDataBeansResourceGroup
- PickBatchInventoryManagersForOrgDisplayPickBatchInventoryDataBeansResourceGroup
- ReceiverOrderManagersForOrgDisplayReceiverOrderManagementDataBeansResourceGroup
- ReturnsAdminOrderManagersForOrgDisplayReturnsAdminOrderManagementDataBeans ResourceGroup
- SuperUserOrderManagersForOrgDisplaySuperUserOrderManagementDataBeans ResourceGroupFulfillmentManagersForOrgDisplayReleaseOrderItemsDatabeanResourceGroup

Order Management

Data resources:

- CustomerOrderManagersForOrgExecuteCustomerServiceOrderWriteCommands OnOrderResource
- CustomerOrderManagersForOrgExecuteCustomerServiceOrderCreateCommands OnStoreEntityResource
- CustomerOrderManagersForOrgExecuteCustomerServiceReturnWriteCommands OnRMAResource
- CustomerOrderManagersForOrgExecuteCustomerServiceReturnCreateCommands OnStoreEntityResource
- CustomerOrderManagersExecuteCustomerWriteCommandsOnUserResource
- CustomerOrderManagersForDefaultOrgExecuteCustomerServiceCustomerWriteCommandsOn UserDataResourceGroupwithGuestRegisterType

Databeans:

- CustomerOrderManagersForOrgDisplayCustomerOrderManagementDatabeans
- MemberOrderManagersForDefaultOrgDisplayGuestMemberDatabeans
- MemberOrderManagersDisplayOrganizationSpecificDatabeans
- MemberOrderManagersDisplayUserDatabeanResourceGroup
- UserOrderManagersForDefaultOrgDisplayGuestMemberDatabeans
- UserOrderManagersDisplayOrganizationSpecificDatabeans
- UserOrderManagersDisplayUserDatabeanResourceGroup
- LogisticsManagersForOrgDisplayOrdersAndReturnsListsDatabeans
- ReturnsManagersForOrgDisplayReturnsListsDatabeans

Payments

Data resources:

- AccountManagersForOrgExecuteAccountCreateCommandsOnOrganizationResource
- AccountAdministratorsForOrgExecuteAccountManageCommandsOnAccountResource
- AccountViewersForOrgExecutePaymentSummaryGenerateCommandsOnAccountResource
- AccountViewersForOrgExecuteStorePaymentAdminCommandsOnStoreEntityResource
- AllUsersExecutePaymentOrderWriteCommandsOnOrderResource

Policy Editor

Data resources:

- StoreAdministratorsForOrgExecuteACPolicyCreateCommandsOnOrganizationResource
- StoreAdministratorsForOrgExecuteACPolicyEditCommandsOnACPolicyResource
- StoreAdministratorsForOrgExecuteACViewPoliciesForUpdateActionsOnOrganizationResource
- StoreAdministratorsForOrgExecuteACViewApplicablePoliciesActionsOnOrganizationResource
- DescendantStoreAdministratorsExecuteACViewPoliciesForOrgActionsOnOrganizationResource

Databeans: StoreAdministratorsForOrgExecuteUserGroupSearchViews

Product Advisor

Databeans:

- ProductAdvisorStatisticiansForOrgDisplayProductAdvisorStatisticsDatabeans
- SalesAssistantStatisticiansForOrgDisplaySalesAssistantStatisticsDatabeans
- ProductAdvisorManagersDisplayPAWCBEDatabeanResourceGroup

- GuidedSellManagersDisplayGSWCBEDatabeanResourceGroup

RFQ

Data resources:

- RFQBuyersExecuteRFQCreateCommandsOnStoreEntityDataResourceGroup
- RFQBuyersManageRFQResourcesTheyOwn
- RFQBuyersManageRFQResponsesForRFQsTheyOwn
- RFQAdministratorsAdministerRFQs
- RFQAdministratorsManageRFQResponses
- RFQSalesManagersForOrgCreateRFQResponse
- RFQSalesManagersExecuteRFQResponseManageCommandsOnRFQResponseResource
- RFQSalesManagersExecuteRFQResponseAdminCommandsOnRFQWithPublicAccess TypeResourceGroup
- RFQSalesManagersExecuteRFQResponseAdminCommandsOnRFQResourceGroup

Databeans:

- RFQBuyersDisplayRFQDataBeanResourceGroupTheyOwn
- RFQBuyersDisplayRFQResponseDataBeansViewabletoRFQOwnerResourceGroup
- RFQSalesViewersDisplayRFQResponseDataBeanResourceGroup
- RFQSalesViewersDisplayRFQDataBeanWithPublicAccessTypeResourceGroup
- RFQSalesViewersDisplayRFQDataBeanResourceGroup

Rules

Data resources: StoreAdministratorsForOrgExecutePersonalizationRuleServiceAdministrationCommands OnStoreEntityResource

Databeans:
StoreAdministratorsForOrgDisplayPersonalizationRuleServiceAdministrationDataBeanResourceGroup

Scheduler

Data resources:

- StoreAdministratorsForOrgExecuteScheduledJobManageCommandsOnStoreEntityResource
- StoreAdministratorsForOrgExecuteScheduledJobManageCommandsOnUserResource

Databeans: StoreAdministratorsForOrgDisplaySchedulerDataBeansResourceGroup

Commerce Accelerator

Data resources:

- B2CCSAViewUsersForOrgExecuteB2CCSAViewActionsOnStoreEntityResource
- B2BCSAViewUsersForOrgExecuteB2BCSAViewActionsOnStoreEntityResource
- CHSCSAViewUsersForOrgExecuteCHSCSAViewActionsOnStoreEntityResource
- RHSCSAViewUsersForOrgExecuteRHSCSAViewActionsOnStoreEntityResource
- CPSCSAViewUsersForOrgExecuteCPSCSAViewActionsOnStoreEntityResource
- RPSCSAViewUsersForOrgExecuteRPSCSAViewActionsOnStoreEntityResource
- HCPCSAViewUsersForOrgExecuteHCPCSAViewActionsOnStoreEntityResource
- MHSCSAViewUsersForOrgExecuteMHSCSAViewActionsOnStoreEntityResource
- MPSCSAViewUsersForOrgExecuteMPSCSAViewActionsOnStoreEntityResource

- SCPCSAViewUsersForOrgExecuteSCPCSAViewActionsOnStoreEntityResource
- SHSCSAViewUsersForOrgExecuteSHSCSAViewActionsOnStoreEntityResource
- SPSCSAViewUsersForOrgExecuteSPSCSAViewActionsOnStoreEntityResource

Shipping

Data resources: ShippingMembershipAdministratorsForOrgExecuteShippingManageCommandsOnStore DataResourceGroup

Databeans: ShippingMembershipAdministratorsForOrgDisplayShippingDatabeanResourceGroup

Taxation

Data resources:
TaxationAdministratorsForOrgExecuteTaxationManageCommandsOnStoreDataResourceGroup

Databeans: TaxationAdministratorsForOrgDisplayTaxationDatabeanResourceGroup

Live Help/ Collaborative Workspaces/Customer Care

Data resources: Live Help:
- LiveHelpAgentsForOrgExecuteLiveHelpRetrieveCommandsOnUserDataResources
- LiveHelpAgentsForOrgExecuteLiveHelpRetrieveCommandsOnOrderDataResources

Data resources: Customer Care:
CustomerCareAdministratorsForOrgExecuteCustomerCareQueueManageCommandsOnStoreResource

Databeans: Live Help: LiveHelpAgentsForOrgDisplayCustomerCareDatabeanResourceGroup

Databeans: Collaborative Workspaces:
CollaborativeWorkspaceAdministratorsForOrgDisplayCollaborativeWorkspaceDatabeanResourceGroup

Store state

Data resources:
- ChannelManagersExecuteStoreStateChangeCommandsOnStoreResource
- AdministrativeRolesForOrgExecuteStoreStateChangeCommandsOnStoreResource
- AdministratorsForOrgAccessStoreWithCloseOrSuspendStateResourceGroup
- AllUsersAccessStoreWithOpenStateResourceGroup

Store Management

Data resources: report delivery:
ReportDeliveryManagersForOrgExecuteSetupReportDeliveryCommandsOnStoreDataResourceGroup

Data resource: store:
- StoreFrontManagersForOrgExecuteStoreFrontRelatedUpdateOnStoreEntityResource
- StoreProfileManagersForOrgExecuteStoreProfileRelatedUpdateOnStoreEntityResource

Default access control policy groups

The default access control policy groups that are shipped with WebSphere Commerce are the following:
- `Management and Administration Policy Group`: This policy group contains all the member management and store administration policies.

- `Guest Shopper Management Policy Group`: This policy group contains all the policies that are related to guest shopper management.
- `Common Shopping Policy Group`: This policy group contains all the shopping-related policies that are common to both consumer direct and B2B scenarios.
- `B2C Policy Group`: This policy group contains all the consumer direct specific shopping policies.
- `B2B Policy Group`: This policy group contains all the B2B specific shopping policies.

Note: The `Management and Administration Policy Group` is the core policy group that should typically apply to all organizations. Whenever an organization subscribes to any policy group, this policy group should also be subscribed. For an organization that owns a store, depending on the type of the store, in addition to the `Management and Administration Policy Group`, it should also subscribe to `Common Shopping Policy Group`, `B2C Policy Group` and `B2B Policy Group`. The `Guest Shopper Management Policy Group` should only be subscribed by the organization that owns guest shoppers, which is the Default Organization in a common scenario.

Notices

Copyright License

Trademarks

Printed in USA