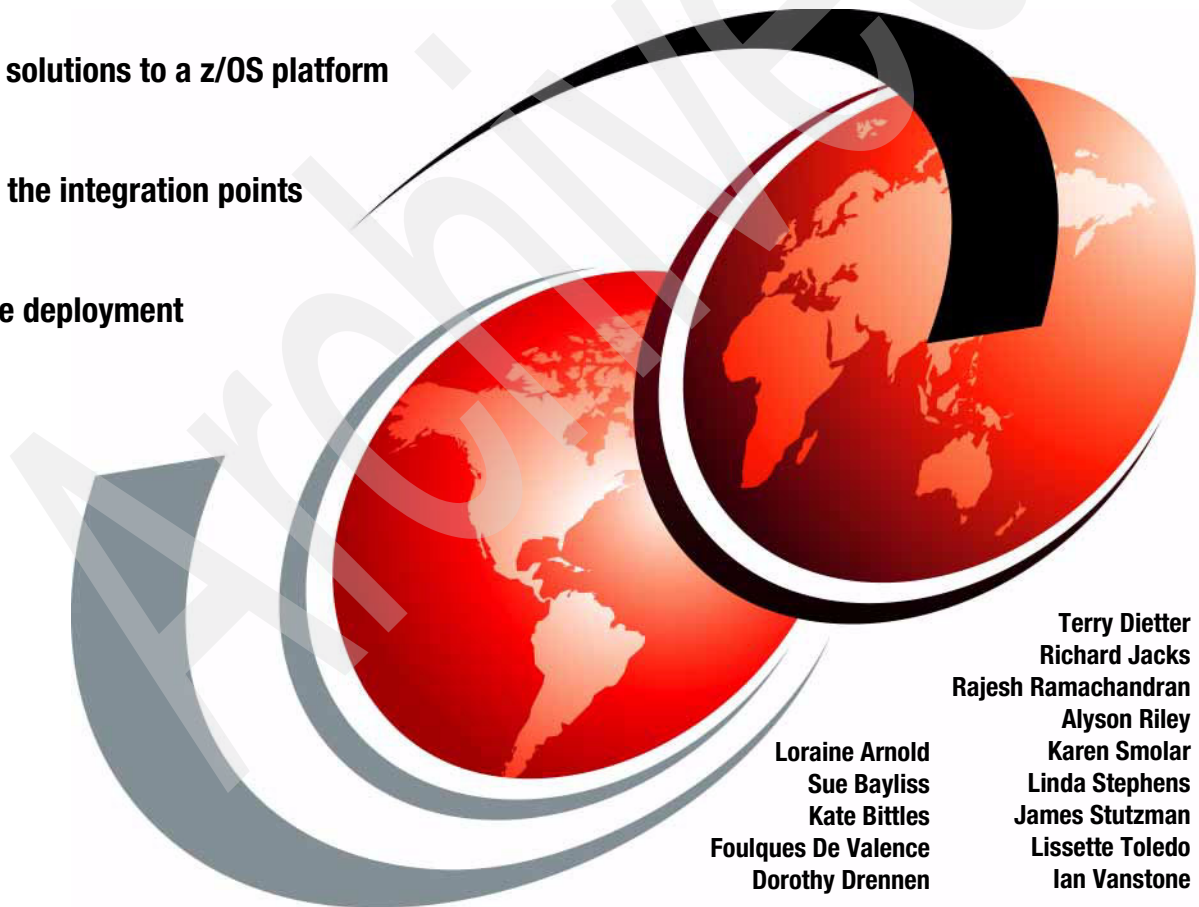


The Mixed Platform Stack Project: Deploying a Secure SOA Solution into z/OS and Mixed z/OS and AIX Environments

Deploy solutions to a z/OS platform

Secure the integration points

Test the deployment



Terry Dietter
Richard Jacks
Rajesh Ramachandran
Alyson Riley
Karen Smolar
Linda Stephens
James Stutzman
Lissette Toledo
Ian Vanstone

Loraine Arnold
Sue Bayliss
Kate Bittles
Foulques De Valence
Dorothy Drennen



International Technical Support Organization

**The Mixed Platform Stack Project: Deploying a
Secure SOA Solution into z/OS and Mixed z/OS
and AIX Environments**

May 2009

Archived

Note: Before using this information and the product it supports, read the information in “Notices” on page ix.

First Edition (May 2009)

This edition applies to IBM WebSphere Application Server V6.1, WebSphere Portal Server V6.1, WebSphere MQ V6.0, WebSphere Message Broker V6.1, WebSphere MQ File Transfer Edition V7, WebSphere Process Server V6.1, WebSphere Service Registry and Repository V6.1, IBM Tivoli Monitoring V6.2, IBM Tivoli Composite Application Manager for WebSphere V6.1, Rational Application Developer V7 and V7.5, WebSphere Integration Developer V6.1, IBM HTTP Server V6.1, IBM DB2 Universal Database V9.1, and IBM CICS Transaction Server V3.2.

© Copyright International Business Machines Corporation 2009. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

| | |
|---|------|
| Notices | ix |
| Trademarks | x |
| Preface | xi |
| The team that wrote this book | xi |
| Become a published author | xiii |
| Comments welcome | xiv |
| Chapter 1. Introduction to the Mixed Platform Stack project | 1 |
| 1.1 Project and scenario overview | 2 |
| 1.2 Project background | 3 |
| 1.3 Project objectives | 3 |
| 1.4 Scenario | 4 |
| 1.5 Scenario realization | 5 |
| 1.5.1 Key features of environments | 8 |
| 1.5.2 Further product-specific details | 11 |
| 1.6 Security configuration | 19 |
| Chapter 2. Application development | 27 |
| 2.1 Application overview | 28 |
| 2.1.1 LGI application flows | 28 |
| 2.1.2 LGI application components | 32 |
| 2.2 Development environment | 39 |
| 2.2.1 Web application development | 39 |
| 2.2.2 Portlet development | 41 |
| 2.3 Securing the application | 42 |
| 2.3.1 Configuring the secure pages in the Web application | 42 |
| 2.3.2 Enabling role-based security in a Web application | 44 |
| 2.3.3 Customizing logic in the J2EE application using role-based security | 44 |
| 2.3.4 Portlet security | 45 |
| 2.4 Securing the development environment | 45 |
| 2.4.1 Enabling administration security using the local operating system .. | 45 |
| 2.4.2 Administration security using Tivoli Directory Server | 46 |
| 2.4.3 Mapping runtime users or groups to application security roles | 46 |
| 2.4.4 Troubleshooting common problems with securing the server | 46 |
| 2.4.5 Rational Application Developer V7.5 and WebSphere Portal Server V6.1 | 47 |
| 2.5 Application testing | 47 |
| 2.5.1 Testing the session and entity beans | 48 |

| | |
|---|-----------|
| 2.5.2 Testing the security roles | 49 |
| Chapter 3. Front-end deployment (AIX, Linux for System z, and z/OS) .. | 51 |
| 3.1 Front-end overview | 52 |
| 3.2 Tivoli security on AIX and Linux for System z | 52 |
| 3.2.1 Installation | 53 |
| 3.2.2 Creating the LDAP structure | 54 |
| 3.3 WebSphere Portal Server 6.1 on AIX and Linux for System z | 55 |
| 3.3.1 Configuration | 56 |
| 3.3.2 Security configuration | 57 |
| 3.3.3 Application deployment | 61 |
| 3.4 WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z | 65 |
| 3.4.1 Creating the configuration | 66 |
| 3.4.2 Configuring security | 66 |
| 3.4.3 Deploying the application | 69 |
| 3.5 WebSphere MQ on AIX and Linux for System z | 70 |
| 3.5.1 Configuring | 70 |
| 3.5.2 Security configuration | 71 |
| 3.5.3 Application deployment | 76 |
| 3.6 DB2 Enterprise 9 on AIX and Linux for System z | 76 |
| 3.6.1 Installing DB2 Enterprise 9 | 76 |
| 3.6.2 Configuration | 77 |
| 3.6.3 Security configuration | 78 |
| 3.6.4 Defining databases | 78 |
| 3.7 IBM HTTP Server 6.1 on AIX | 78 |
| 3.7.1 Configuration | 78 |
| 3.7.2 Security configuration | 79 |
| 3.8 WebSphere Application Server 6.1 Network Deployment on z/OS | 80 |
| 3.8.1 Configuration | 81 |
| 3.8.2 Tools to assist in configuration | 82 |
| 3.8.3 Security configuration | 82 |
| 3.8.4 Application deployment | 85 |
| 3.8.5 JDBC resource definitions | 86 |
| 3.9 WebSphere MQ on z/OS | 88 |
| 3.9.1 Configuration | 88 |
| 3.9.2 Security configuration | 89 |
| 3.9.3 Application deployment | 93 |
| 3.10 DB2 Enterprise 9 on z/OS | 93 |
| 3.10.1 The LGI secure application configuration | 94 |
| 3.11 WebSphere MQ File Transfer Edition on AIX | 94 |
| 3.11.1 Configuration of an AIX agent | 94 |
| 3.11.2 Security configuration | 95 |

| | | |
|-------------------|---|-----------|
| 3.11.3 | Transferring files | 95 |
| 3.11.4 | Debugging WebSphere MQ FTE problems | 96 |
| 3.11.5 | Application deployment | 96 |
| 3.12 | IBM Tivoli Composite Application Manager on AIX | 96 |
| 3.12.1 | Configuration of the AIX agents and data collectors | 96 |
| 3.12.2 | Application deployment | 97 |
| Chapter 4. | Deployment of the back-end components | 99 |
| 4.1 | Overview of the back-end components | 100 |
| 4.2 | WebSphere Application Server 6.1 on z/OS | 100 |
| 4.2.1 | Configuration | 100 |
| 4.2.2 | Security configuration | 101 |
| 4.2.3 | Application deployment | 102 |
| 4.3 | WebSphere Process Server on z/OS | 103 |
| 4.3.1 | Configuration | 103 |
| 4.3.2 | Service integration configuration | 104 |
| 4.3.3 | Security configuration | 106 |
| 4.3.4 | WebSphere Process Server security | 108 |
| 4.3.5 | Application deployment | 109 |
| 4.4 | WebSphere Service Registry and Repository on z/OS | 110 |
| 4.4.1 | Configuration | 110 |
| 4.4.2 | Security configuration | 111 |
| 4.4.3 | Application deployment | 112 |
| 4.5 | WebSphere MQ on z/OS | 113 |
| 4.5.1 | Configuration | 113 |
| 4.5.2 | Security configuration | 115 |
| 4.5.3 | Application deployment | 121 |
| 4.6 | WebSphere Message Broker on z/OS | 122 |
| 4.6.1 | Configuration | 122 |
| 4.6.2 | Security configuration | 123 |
| 4.6.3 | SOA messaging flow development | 125 |
| 4.6.4 | Application deployment | 125 |
| 4.7 | DB2 Enterprise 9 on z/OS | 126 |
| 4.7.1 | DB2 data sharing | 126 |
| 4.8 | CICS Transaction Server v3.2 for z/OS | 129 |
| 4.8.1 | Configuration | 129 |
| 4.8.2 | Security configuration | 130 |
| 4.8.3 | Application deployment | 130 |
| 4.9 | WebSphere MQ File Transfer Edition on AIX | 130 |
| 4.9.1 | Configuring the AIX coordination queue manager | 131 |
| 4.9.2 | Security configuration | 131 |
| 4.9.3 | Debugging WebSphere MQ FTE problems | 132 |
| 4.9.4 | Application deployment | 132 |

| | |
|---|------------|
| 4.10 WebSphere MQ File Transfer Edition on z/OS | 132 |
| 4.10.1 Configuration of the z/OS agent | 132 |
| 4.10.2 Security configuration | 133 |
| 4.10.3 Debugging WebSphere MQ FTE problems | 133 |
| 4.10.4 Application deployment | 134 |
| 4.11 IBM Tivoli Monitoring on AIX | 134 |
| 4.11.1 Configuration of IBM Tivoli Monitoring | 134 |
| 4.11.2 Application deployment | 134 |
| Chapter 5. Securing integration points | 135 |
| 5.1 Web Services Security on AIX and z/OS | 136 |
| 5.2 Configuration of the runtime environments | 137 |
| 5.3 Security configuration | 137 |
| 5.3.1 Using public-key cryptography to secure Web services | 138 |
| 5.3.2 Configuring WS-Security in WebSphere Application Server | 139 |
| 5.3.3 Configuring WS-Security in WebSphere Message Broker | 140 |
| 5.4 LGI scenario implementation information | 141 |
| 5.4.1 Configuring WS-Security for the MVR Web service call | 141 |
| 5.4.2 Configuring WS-Security for the underwriter Web service call | 141 |
| Chapter 6. Testing the solution | 143 |
| 6.1 Test methodology | 144 |
| 6.1.1 Approach by the z/OS Integration Test Team | 144 |
| 6.1.2 Test scenarios | 146 |
| 6.2 Test results | 148 |
| Appendix A. Open issues | 149 |
| ClassCastException customizing WS-Security bindings in WebSphere Process Server 6.1 | 150 |
| No predefined keystores customizing WS-Security bindings in WebSphere Application Server 6.1 | 151 |
| Messaging Engines do not start after upgrading WebSphere Application Server to 6.1.0.18 | 151 |
| Appendix B. Detailed instructions | 153 |
| Instructions for 2.3, 'Securing the application' | 154 |
| Enabling role-based security in a Web application | 155 |
| Instructions for 2.4, 'Securing the development environment' | 157 |
| Enabling administration security using the local operating system | 157 |
| Instructions for 2.5, 'Application testing' | 158 |
| Testing the session and entity beans | 158 |
| Testing the security roles | 159 |
| Instructions for 3.2, 'Tivoli security on AIX and Linux for System z' | 160 |
| Installation | 160 |

| | |
|--|-----|
| Instructions for 3.4, 'WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z' | 169 |
| Configuring security | 169 |
| Using SSL between LDAP and WebSphere Application Server | 174 |
| Using SSL between the Web server and WebSphere Application Server | 179 |
| Application deployment | 183 |
| Instructions for 3.5, 'WebSphere MQ on AIX and Linux for System z' | 183 |
| Configuring security | 184 |
| Troubleshooting | 192 |
| Instructions for 3.6, 'DB2 Enterprise 9 on AIX and Linux for System z' | 192 |
| Database definition | 193 |
| Instructions for 3.7, 'IBM HTTP Server 6.1 on AIX' | 194 |
| Configuring security | 194 |
| Instructions for 3.9, 'WebSphere MQ on z/OS' | 195 |
| Configuring access control | 195 |
| Configuring Secure Sockets Layer | 196 |
| Exchanging certificates with another z/OS queue manager | 198 |
| Instructions for 3.11, 'WebSphere MQ File Transfer Edition on AIX' | 199 |
| Configuring an AIX agent | 199 |
| Transferring files | 200 |
| Instructions for 3.12, 'IBM Tivoli Composite Application Manager on AIX' | 201 |
| Configuring Tivoli Enterprise Management Agent | 201 |
| Configuring WebSphere Data Collector | 201 |
| Verifying the installation by using Tivoli Enterprise Portal | 202 |
| Instructions for 4.3, 'WebSphere Process Server on z/OS' | 204 |
| Configuring security | 204 |
| Instructions for 4.4, 'WebSphere Service Registry and Repository on z/OS' | 210 |
| Configuring security | 210 |
| SOA messaging WSDL | 212 |
| Instructions for 4.5, 'WebSphere MQ on z/OS' | 213 |
| Configuring security | 213 |
| Useful RACDCERT commands | 220 |
| Troubleshooting WebSphere MQ SSL configuration errors | 220 |
| Instructions for 4.6, 'WebSphere Message Broker on z/OS' | 222 |
| Configuring security | 222 |
| Instructions for 4.9, 'WebSphere MQ File Transfer Edition on AIX' | 225 |
| Configuring the AIX coordination queue manager | 226 |
| Instructions for 4.10, 'WebSphere MQ File Transfer Edition on z/OS' | 226 |
| Configuring the z/OS agent | 227 |
| Instructions for 4.11, 'IBM Tivoli Monitoring on AIX' | 228 |
| Installing DB2 Enterprise Server Edition V8.1 | 228 |
| Installing Fix Pack 16 | 229 |
| Configuring IBM Tivoli Monitoring | 230 |

| | |
|--|-----|
| Creating the DB2 Data Warehouse database | 232 |
| Starting Tivoli Enterprise Management Server and Tivoli Enterprise Portal Server | 233 |
| Configuring the Warehouse Proxy agent | 233 |
| Starting the Tivoli Enterprise Portal Server | 234 |
| Configuring the Summarization and Pruning agent | 234 |
| Verifying the installation | 235 |
| Installing IBM Tivoli Monitoring application support for IBM Tivoli Composite Application Manager for WebSphere | 236 |
| Instructions for 5.1, 'Web Services Security on AIX and z/OS' | 236 |
| Preparation | 236 |
| Configuring WS-Security | 237 |
| Algorithms | 239 |
| LGI scenario implementation information | 240 |
| Application deployment | 249 |
| Troubleshooting | 251 |
| Known issues | 252 |
| Related publications | 253 |
| IBM Redbooks | 253 |
| Other publications | 253 |
| Online resources | 253 |
| How to get Redbooks | 256 |
| Help from IBM | 256 |

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.


COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

| | | |
|-------------------------|---|------------|
| AIX® | OMEGAMON® | System p® |
| CICSplex® | Parallel Sysplex® | System z9® |
| CICS® | Passport Advantage® | System z® |
| DB2 Universal Database™ | pSeries® | Tivoli® |
| DB2® | RACF® | WebSphere® |
| developerWorks® | Rational® | z/OS® |
| eServer™ | Redbooks® | z9® |
| IBM® | Redbooks (logo)  ® | zSeries® |

The following terms are trademarks of other companies:

Interchange, and the Shadowman logo are trademarks or registered trademarks of Red Hat, Inc. in the U.S. and other countries.

EJB, Enterprise JavaBeans, J2EE, Java, JavaBeans, JavaServer, JDBC, JSP, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Internet Explorer, Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

The IBM® System z® platform is the strategic core of business world wide. By using a realistic customer scenario, two IBM teams ventured to demonstrate how to deploy the IBM service-oriented architecture (SOA) portfolio on IBM z/OS® and on z/OS in partnership with additional platforms such as AIX® and Linux® for System z. The teams created the experience that is documented in this IBM Redbooks® publication to explain the work that is required to create, deploy, and test the SOA solution on both z/OS and z/OS with additional platforms. The teams also performed extensive testing to verify the correct behavior of the platforms, products, and applications involved.

This Redbooks publication covers the product configuration that is necessary to build the SOA solution described in the project scenario. This book provides useful hints and tips that were discovered during the course of testing to ensure successful solution deployment. It also provides an extensive set of references to other documents that proved useful for building the solution.

This book is designed for IT professionals who are interested in creating an SOA solution either entirely on z/OS or on z/OS in conjunction with other platforms. Prior to reading this book, you must have basic knowledge of SOA solutions, z/OS or other platforms, and the SOA products running on those platforms.

The team that wrote this book

This book was produced by a team of specialists from around the world:

Loraine Arnold is an Advisory Programmer whose IBM career started in 1982. Her experience includes systems programming in support of MVS, IBM IMS, DB2®, WebSphere® MQ, and WebSphere Application Server. She has also worked in the software development area as a developer, tester, and change team support person. Loraine currently works in the z/OS Integration Test area where she supports application development for I/T workloads using IBM WebSphere Process Server, WebSphere Application Server, WebSphere Service Registry and Repository, WebSphere MQ, DB2, and IMS.

Sue Bayliss is a Software Engineer who joined IBM in 1990. She has focused on software testing, performing solution and integration testing centered around the WebSphere brand.

Kate Bittles is an Accredited IT Architect with experience in services engagements and product development, specializing in messaging. She currently manages the FIT team in Hursley. She has been with IBM since 1995.

Foulques De Valence is System z Web and Security IT Architect currently working on the STG Lab Services team. He provides consulting services worldwide about SOA, security, and WebSphere products with a focus on System z infrastructures. Previously, Foulques was a Web infrastructure IT Architect in France specializing in SOA and z/OS.

Dorothy Drennen is a Senior Project Manager currently working with the Federated Integration Test (FIT) and IBM Software Group (SWG) Consumability teams. Dorothy joined IBM in 1998 and has experience in service engagements, product development management, and program management.

Terry Dietter is an Advisory Programmer who joined IBM in 1988. She has worked on many projects including software development supporting MVS and VM. She also worked in software test for AIX as a tester and Storage Area Network administrator. Terry currently works in the Linux for System z Distribution Testing area.

Richard Jacks is a Senior IT Specialist and has been with IBM since 2001. He began his IBM career in the SWG Lab Services team where he specialized in pervasive technologies and later WebSphere Portal and related technologies. Richard currently works on the FIT team.

Rajesh Ramachandran is a Senior IT Architect in IBM Systems and Technology Group System z Lab Services. Rajesh has 14 years of experience in designing applications and infrastructure across various platforms, including mainframe, UNIX®, and Linux.

Alyson Riley is a Senior Software Engineer and Information Architect currently working in the IBM Corporate Information Development organization. She began her IBM career in 1997 in STG and is now responsible for information and user experience strategy for the corporate information development community.

Karen Smolar is a Senior Software Engineer with 20 years experience with IBM. She has held various positions in application development, test, system architecture, and consulting. Karen is currently a member of the z/OS Integration Test team. She is responsible for test strategy and testware development using SOA infrastructure and solutions.

Linda Stephens is an IT specialist with the FIT team in Hursley, UK, where she specializes in WebSphere products, particularly WebSphere Application Server, WebSphere Process Server, and Monitor.

James Stutzman is a Software Test Specialist, currently working in the z/OS Integration Test area, doing system administration for DB2 on z/OS.

Lisette Toledo is a Software Test Engineer who joined IBM in 2004. She currently works in the z/OS Integration Test area, doing system administration for WebSphere MQ and WebSphere Message Broker in z/OS.

Ian Vanstone is an Advisory Software Engineer who joined IBM in 2000. He spent eight years working in WebSphere MQ development, focusing on intercommunication and availability messaging architectures on z/OS and distributed platforms. Ian recently moved to his current position as an integration test specialist on the FIT team.

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- ▶ Send your comments in an e-mail to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Introduction to the Mixed Platform Stack project

This chapter provides an overview of the Mixed Platform Stack project and the scenario that is used as the basis for the project. It includes the following topics:

- ▶ “Project and scenario overview” on page 2
- ▶ “Project background” on page 3
- ▶ “Project objectives” on page 3
- ▶ “Scenario” on page 4
- ▶ “Scenario realization” on page 5
- ▶ “Security configuration” on page 19

1.1 Project and scenario overview

The Mixed Platform Stack project demonstrated deployment of the major components of a service-oriented architecture (SOA) solution on the z/OS platform (or on a distributed platform in addition to z/OS) with little or no modification of applications. The project was a joint effort between the IBM Software Group (SWG) Federated Integration Test (FIT) and the IBM Storage Group (STG) z/OS Integration Test teams.

In this project, the teams considered the key integration points of products running on different platforms and tested them to ensure interoperability. In addition, the project included a particular focus on security configuration. The project verified that the platforms, products, and applications involved in the scenario can be deployed successfully in both environments. The team also conducted extensive tests to verify the correct behavior of the platforms, products, and applications.

This IBM Redbooks publication documents the experiences and results that the teams discovered while deploying an SOA solution into both an IBM System z environment and a mixed environment that included the IBM System z and System p® platforms. This book contains specific steps to deploy a scenario to both environments, as described in the following chapters:

- ▶ Chapter 1, “Introduction to the Mixed Platform Stack project”, provides background information about the scenario that was used as the basis for this project.
- ▶ Chapter 2, “Application development” on page 27, describes the application that was used in this scenario and the tools and methods employed to develop the application.
- ▶ Chapter 3, “Front-end deployment (AIX, Linux for System z, and z/OS)” on page 51, describes the components of the scenario that provide the user interface. It covers the deployment of these components into AIX, Linux for System z, and z/OS.
- ▶ Chapter 4, “Deployment of the back-end components” on page 99, describes the products and considerations that are specific to the z/OS components.
- ▶ Chapter 5, “Securing integration points” on page 135, describes the methods that are used to secure the integration points between products. This chapter has a particular focus on Web Services Security (WS-Security).
- ▶ Chapter 6, “Testing the solution” on page 143, outlines an approach to testing that can be used to verify the correct deployment and operation of the scenario.

- ▶ Appendix A, “Open issues” on page 149, contains a list of open issues that are relevant to the work described in this book and that IBM is currently investigating.
- ▶ Appendix B, “Detailed instructions” on page 153, contains detailed instructions that correspond to the chapters in this book. While the chapters provide a high-level view of the activities that the teams performed to achieve the goals of this project, this appendix provides the low-level implementation procedures.

1.2 Project background

The IBM Software Group FIT team is a worldwide team whose mission is to test the integration of IBM products and provide feedback to IBM product teams, customer-facing teams, and customers. The FIT team uses realistic customer-type scenarios to perform integration testing on stacks of products.

The IBM Systems and Technology Group z/OS Integration Test team, also known as zPET, is responsible for the final phase of System z product testing. The team’s mission is to validate z/OS functionality in an environment that closely simulates realistic production workloads and focuses on typical customer tasks.

Early in 2008, the FIT team began working in conjunction with the zPET team on the Mixed Platform Stack Project. The FIT team was responsible for deploying and testing the solution in a mixed platform environment that included z/OS and AIX. zPET was responsible for deploying and testing the solution in a z/OS-only environment. During the second half of 2008, Linux for System z was added to the zPET environment, providing an alternative platform for components of the front-end deployment.

1.3 Project objectives

The Mixed Platform Stack project included the following high-level objectives:

- ▶ Create and maintain a secure solution on the customer’s platform combination of choice (z/OS, AIX, and Linux for System z).
- ▶ Ensure that all products in the solution work well together, regardless of platform specifics.
- ▶ Demonstrate that SOA solutions can be easily deployed on any platform and can take advantage of platform-specific capabilities.

The project successfully demonstrated that an SOA solution can be deployed to the z/OS platform and z/OS in combination with other platforms. The scenario used both J2EE™ and existing technologies and demonstrated the integration between these technologies on multiple platforms. In summary, this project demonstrated that the platform and software are flexible, allowing customers to choose technologies that best meet their needs.

1.4 Scenario

The LGI scenario used for the test activities simulates a motor insurance quote system. The scenario is based on the result of a merger between two fictional insurance companies, Lord General Insurance (LGI) and Direct Car (DC). LGI is a large, established mainframe-based company that acquired DC. DC is an insurance newcomer with Internet-based IT skills and infrastructure.

Following the merger, the company implemented an insurance quote and policy system based on SOA. The quote and policy system included the following features:

- ▶ A direct Internet channel for users
 - ▶ An enterprise service bus (ESB) to transform and route messages to the two independent back-end systems and to transfer files to the back-end systems
- The dual back-end systems provide the additional technical challenge of integrating different technologies.
- ▶ A service registry for looking up endpoints and routing service requests
 - ▶ Two distinct back-end systems (DC and LGI), both capable of performing the business logic required for the following purposes:
 - Issue insurance quotes and policies
 - Store customer and policy information relevant to those policies

Note: Data protection issues required that the customer data held in each of the LGI and DC back-end systems remain separate, with its own access controls.

- ▶ A common business process to handle the offline background checks required for final acceptance of an insurance policy
- ▶ Web services to perform specific application functions
- ▶ Monitoring of applications, products, and systems

Figure 1-1 illustrates the logical architecture used in the LGI scenario. The arrows indicate the direction of request flows. All calls are two-way request/response.

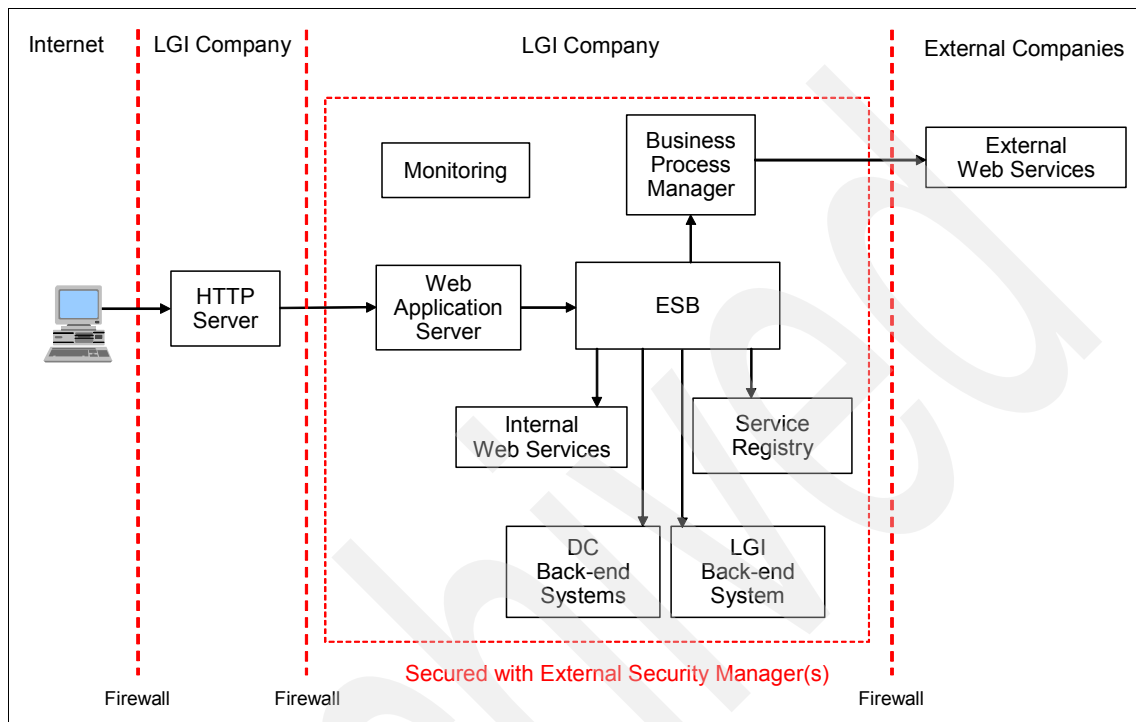


Figure 1-1 LGI scenario logical architecture

1.5 Scenario realization

Two implementations or realizations of the LGI scenario were used in this project:

- ▶ A mixed stack environment distributed across AIX, Microsoft® Windows®, and z/OS
- ▶ A z/OS and Linux for System z environment

Each scenario realization has two alternative front-end implementations. These front-end implementations demonstrate the different technologies used to host a Web application that provides the browser-based user interface:

- ▶ WebSphere Application Server hosting JavaServer™ Pages (JSP™) and servlets
- ▶ WebSphere Portal Server hosting portlets

Figure 1-2 shows the application components that were deployed in the AIX and z/OS mixed platform environment. All flows are two-way request/response, where the arrows indicate the direction of the request flow.

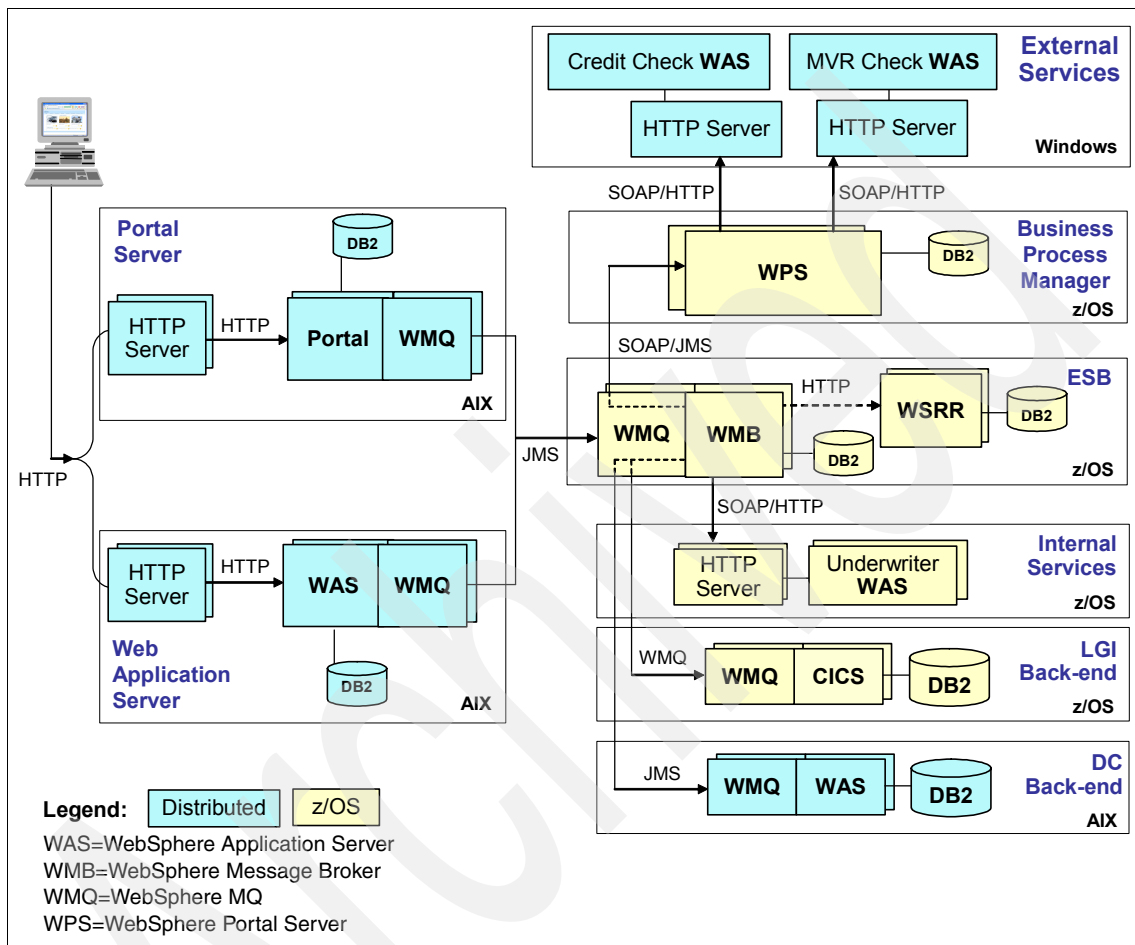


Figure 1-2 AIX, Windows, and z/OS environments

Figure 1-3 shows the application components that we deployed in the z/OS platform environment. All flows are two-way request/response, where the arrows indicate the direction of the request flow.

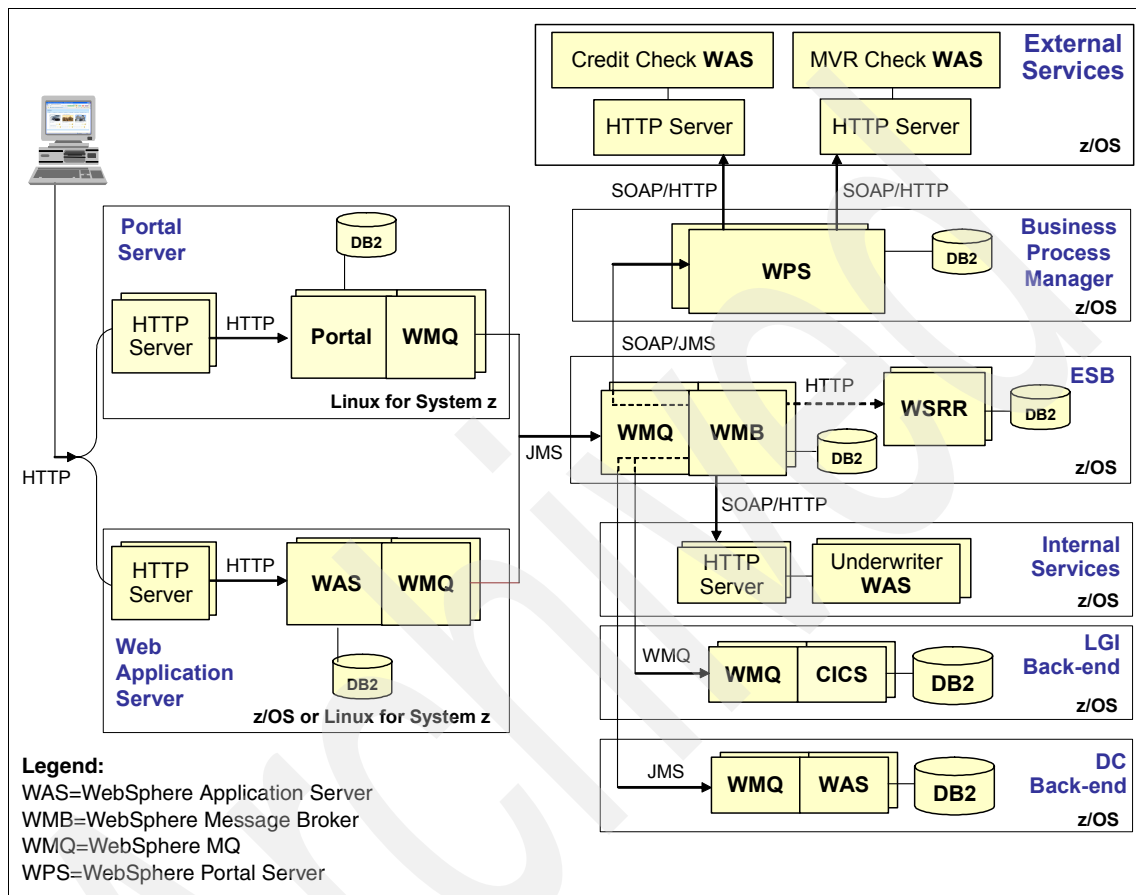


Figure 1-3 z/OS and Linux for System z environment

Key integration points

The LGI scenario environment exercises the product integration points listed in Table 1-1.

Table 1-1 Product integration points in the LGI environment

| Product acting in the client role | Product acting in the server role | Technology used for integration |
|-----------------------------------|--|--|
| IBM HTTP Server | WebSphere Application Server | HTTP between the Web server plug-in and the WebSphere Application Server Web container |
| WebSphere Application Server | WebSphere Message Broker | Java™ Message Service (JMS) over WebSphere MQ cluster channels |
| WebSphere Portal Server | WebSphere Message Broker | JMS over WebSphere MQ cluster channels |
| WebSphere Message Broker | WebSphere Service Registry and Repository (WSRR) | HTTP to Service Registry application |
| WebSphere Message Broker | WebSphere Application Server Web service (Underwriter) | SOAP/HTTP Web service request |
| WebSphere Message Broker | DC back-end server (WebSphere MQ/WebSphere Application Server) | JMS over WebSphere MQ channels |
| WebSphere Message Broker | LGI back-end server (WebSphere MQ-CICS Bridge) | WebSphere MQ over WebSphere MQ cluster channels |
| WebSphere Message Broker | WebSphere Process Server | SOAP/JMS Web service request via WebSphere MQ link to the Business Process Container (BPC) service integration bus |

1.5.1 Key features of environments

The solution used in the test activities is designed for availability, reliability, and scalability. On z/OS, the solution takes advantage of Parallel Sysplex® capabilities within each product. With the Parallel Sysplex capabilities, any component can be run on any available logical partition (LPAR) and with the highest quality of service.

In particular, the environments in the project include the following products that were configured to optimize availability, reliability, and scalability:

- ▶ WebSphere Application Server, WebSphere Portal Server, WebSphere Process Server, and WebSphere Service Registry and Repository clusters provide workload balancing and failover.
- ▶ DB2 data-sharing on z/OS provides maximum availability and reliability.
- ▶ A combination of WebSphere MQ features in the LGI solution provide the most robust, secure, and highly available environment possible. WebSphere MQ shared queues are used on z/OS to provide the highest availability and scalability. WebSphere MQ clusters are used across the range of platforms to simplify administration, provide workload balancing, and improve availability.
- ▶ Multiple WebSphere Message Brokers in a broker domain, using WebSphere MQ clustering and shared queue facilities, provide high availability and workload balancing.
- ▶ CICSplex® processing provides the highest availability, scalability, and manageability for the back-end components.

These implementations use a variety of IBM products. The levels of software listed in following tables indicate the versions and releases of the products used during this project. The PTFs and fix packs that are listed are required to address issues that the test teams discovered during this particular test. However, the tables do not include a complete list of services that might have been applied to the environments for other reasons.

Table 1-2 lists the IBM application development products used in the scenario realization.

Table 1-2 IBM application development products used in the scenario realization

| Product | Version | Required service level | Platform |
|----------------------------------|---------|--|--------------|
| Rational® Application Developer | 7.0.0.6 | For JSP, servlets and Enterprise JavaBeans™ (EJB™) | Windows XP |
| | 7.5 | For portlets | Windows XP |
| WebSphere Message Broker Toolkit | 6.1.0.2 | | Windows 2003 |
| WebSphere Integration Developer | 6.1.0.1 | Build id: 6.1.0.1ifix001-20080617_1257 | Windows XP |

Table 1-3 lists the runtime products used in the scenario realization.

Table 1-3 IBM runtime products used in the scenario realization

| Product | Version | Required service level | Platform |
|---|---------|---|--------------------|
| IBM CICS® Transaction Server for z/OS | 3.2 | | z/OS |
| IBM DB2 Universal Database™ Enterprise Server Edition (DB2) | 9.1 | Fix Pack 4 (9.1.4) | AIX |
| IBM DB2 Universal Database Enterprise Server Edition for z/OS (DB2) | 9.1 | Including PTF PK59069 | z/OS |
| IBM HTTP Server | 6.1 | Fix Pack 15 (6.1.0.15) | AIX |
| | 6.1 | Fix Pack 15 (6.1.0.15) | Windows |
| IBM WebSphere Application Server Network Deployment (ND) | 6.1 | Fix Pack 15 (6.1.0.15) | AIX |
| | 6.1 | Fix Pack 15 (6.1.0.15) | Windows |
| | 6.1 | Fix Pack 18 (6.1.0.18) | z/OS |
| IBM WebSphere Message Broker for z/OS | 6.1 | Fix Pack 6.1.0.2 | z/OS |
| IBM WebSphere MQ | 6.0 | Fix Pack 6.0.2.4 | AIX |
| | 6.0 | Fix Pack 6.0.2.4 | Linux for System z |
| | 6.0 | Fix Pack 6.0.2.5 including PTF UK22256 | z/OS |
| | 7.0 | | AIX |
| IBM WebSphere MQ File Transfer Edition (FTE) | 7.0 | | AIX |
| | 7.0 | | z/OS |
| WebSphere Portal Server | 6.1 | | AIX |
| | 6.1 | | Linux for System z |
| WebSphere Process Server for z/OS | 6.1 | Fix Pack 1 (6.1.0.1) (based on WebSphere Application Server 6.1.0.18) | z/OS |

| Product | Version | Required service level | Platform |
|--|---------|--|----------|
| WebSphere Service Registry and Repository for z/OS | 6.1 | Based on WebSphere Application Server 6.1.0.18 | z/OS |
| IBM Tivoli® Monitoring (obtained as part of IBM Tivoli Composite Application Manager for SOA Platform) | 6.2 | Fix Pack 1 (6.2.0.1) | AIX |
| IBM Tivoli Composite Application Manager for WebSphere (obtained as part of IBM Tivoli Composite Application Manager for SOA Platform) | 6.1 | Fix Pack 3 (6.1.0.3) and iFix 12 | AIX |

1.5.2 Further product-specific details

The following sections provide business reasons for product use and related architectural details.

SOA messaging

With the complex set of products and services to manage in the LGI scenario environment, there is an increasing need for a centralized point of control and accountability to provide governance capabilities and improve business agility. The LGI scenario environment has a strategic vision for a more holistic approach to service management, regardless of underlying implementation. This vision is supported by emerging products and technologies. A recent evolutionary step in the world of SOA and messaging is the exposure of WebSphere MQ assets (for example, queues and applications) as SOA services. Because the LGI scenario is an SOA solution built upon a WebSphere MQ messaging network, there is a good fit here between existing infrastructure, strategic vision, and newly available architectural patterns.

In recent years, organizations have begun to define WebSphere MQ assets as services, but with no clear standard for describing them. As a consequence, the various implementations adopted do not follow a common model and, therefore, are mostly incompatible. In response to the emerging requirements in this domain, WebSphere MQ SupportPac MA93 (WebSphere MQ Service Definition) was released in November 2007. MA93 is a clear standard that helps promote standardization around service modeling, deployment processes, and skills. Although the definitions in MA93 are not open standards, they have been (and continue to be) developed based on customer feedback, Web services standards, and SOA best practices.

Defining WebSphere MQ assets as SOA services provides the following benefits:

- ▶ Dynamic lookup
 - For request routing and service mobility
 - For policy processing (for example, for dynamic Quality of Service (QoS) changes)
- ▶ WebSphere Service Registry and Repository governance
 - Centralized repository
 - Impact analysis (for example, which services are being used and which queue managers and queues are involved)
 - Control of the application life cycle (for example, plan, manage, version, and deprecate)
- ▶ Preparing a base for future developments in this evolving area

The immediate priorities for the LGI scenario environment are request routing, centralized repository, and impact analysis. However, the adoption of the common architecture that supports our priority items will provide a solid foundation for future adoption of other items, including policy processing and lifecycle control.

A common pattern within the area of SOA messaging is the combined use of WebSphere MQ, WebSphere Message Broker, and WebSphere Service Registry and Repository. In the pattern, WebSphere MQ provides the service endpoints. WebSphere Service Registry and Repository holds Web Services Description Language (WSDL) that describes the WebSphere MQ service endpoints. WebSphere Message Broker provides endpoint lookup, policy processing, routing, and transformation. The routing carried out by WebSphere Message Broker includes the routing of messages to WebSphere MQ queues based on information queried from the WSDL held by WebSphere Service Registry and Repository.

The pattern enhances the SOA credentials of the LGI scenario and fits naturally into the stack, given that all three products are already used in the back-end deployment. Therefore, the pattern will be deployed in the LGI scenario environment to provide request routing, centralized repository, and impact analysis.

We focused on the following WebSphere MQ-invoked back-end services:

- ▶ LGI back-end service
- ▶ DC back-end service

Any routing of messages to these back-end services involves a service endpoint lookup to route messages to an appropriate destination. Routing messages to

reply queue destinations does not involve endpoint lookup. As we roll out new versions of the services, or expand or consolidate the services in the future, WebSphere Service Registry and Repository provides the central point of control and enablement for service changes as part of an improved service lifecycle model. This model can be used for other non-WebSphere MQ services, giving a consistent service exposure model regardless of service technology.

It is possible to take the exposure of WebSphere MQ assets as services a step further by forcing all queue usage to be preceded by an endpoint lookup. This provides a more rigid governance model for service consumers and service providers. However, this additional step was not deemed necessary for the LGI scenario because of the following reasons:

- ▶ WebSphere Message Broker MQInput nodes queues cannot be set based on an endpoint lookup. Although it is trivial and often preferable to set MQInput queue names manually during message flow deployment, requirements for deploy time endpoint lookups are emerging.
- ▶ The benefits of defining reply queue destinations depend on the system requirements. Reply queues can be specified in three places:
 - The request message
 - The WSDL in the following two locations:
 - In the service port `wmqservice:address` `wmq` Internationalized Resource Identifier (IRI)
 - In the service port `wmqservice:address` `replyTo` field

If a service provider honors a WSDL specified reply queue destination, this effectively hard codes the service provider to reply to a single destination. This conflicts with established best practices of specifying reply queue destinations in the request message so that service providers can use that information to reply to any service requester located at any destination. However, when considering the benefits to be gained from storing queue information in WSDL stored in WSRR (for example, to aid interactions with development tooling and provide impact analysis), there is reason to store reply queue destinations in WSDL in some cases.

- ▶ Despite a clear standard to defining WebSphere MQ assets as services being available in SupportPac MA93, this area is new and subject to developments. Therefore, a step-by-step approach is seen as beneficial to respond better to emerging best practices.

For other organizations, performance might be a concern, although the caching features provided by WebSphere Message Broker and WebSphere Service Registry and Repository reduce this concern substantially.

The deployment of the SOA messaging features described previously crosses WebSphere MQ, WebSphere Message Broker, and WebSphere Service Registry and Repository as follows:

- ▶ Creation of WSDL to define the WebSphere MQ service endpoints (see 4.4, “WebSphere Service Registry and Repository on z/OS” on page 110)
- ▶ Loading WSDL into WebSphere Service Registry and Repository (see 4.4, “WebSphere Service Registry and Repository on z/OS” on page 110)
- ▶ Creation of WebSphere Message Broker message flows (see 4.6, “WebSphere Message Broker on z/OS” on page 122)
- ▶ Deployment of WebSphere Message Broker message flows (see 4.6, “WebSphere Message Broker on z/OS” on page 122)

See the following resources for more information about exposing WebSphere MQ queues and applications as SOA services:

- ▶ *Take Control of Your MQ Applications with IBM WebSphere Service Registry and Repository!*
ftp://ftp.software.ibm.com/software/integration/wsrr/Take_control_of_your_MQ_applications.pdf
- ▶ Supportpac MA93
http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24017518&loc=en_US&cs=utf-8&lang=en
- ▶ *Accessing WebSphere Service Registry and Repository using WebSphere Message Broker V6.1 nodes*
http://www.ibm.com/developerworks/websphere/library/techarticles/0806_crocker/0806_crocker.html
- ▶ The “Manual Endpoints” topic in the WebSphere Service Registry and Repository 6.2 Information Center, which provides useful information about support and tooling for administration of WebSphere MQ endpoints from the WebSphere Service Registry and Repository administration console
http://publib.boulder.ibm.com/infocenter/sr/v6r2/index.jsp?topic=/com.ibm.sr.doc/rwsr_profiles_governanceprofile_3_2.html
- ▶ Further information for WebSphere MQ 7.0.0.1 support for WSDL creation, in the WebSphere MQ Explorer interface

WebSphere MQ File Transfer Edition

The existing WebSphere MQ messaging backbone of the LGI scenario environment is used easily to transfer files with WebSphere MQ FTE. WebSphere MQ FTE provides reliable transfer, audit capabilities, and easy administration, without requiring application code.

In addition to the main request quote and accept quote Web applications, a WebSphere Portal Server portlet enables existing LGI customers to upload graphics files of accident scenes, which are used to assess claims. The portlet (deployed on all front-end servers) stores the graphics files on the local machine, and WebSphere MQ FTE transfers the files to a back-end system.

WebSphere MQ FTE availability: At the time of writing WebSphere MQ FTE was not available on Linux for System z. Therefore, it was only deployed on AIX and z/OS.

WebSphere MQ FTE transfers files between two or more WebSphere MQ FTE agents. These agents are connected to WebSphere MQ queue managers by using either a client or bindings connection. In the LGI scenario, files are transferred from all front-end machines to a single back-end machine. Because these machines already host queue managers, a WebSphere MQ FTE agent is connected to each front-end queue manager, and a WebSphere MQ FTE agent is connected to the back-end queue manager (all using bindings connections). Although the back-end queue manager can use shared queues, WebSphere MQ FTE does not support multiple agents using a single set of shared agent queues. Therefore, the z/OS agent only runs on one LPAR.

In addition to agents and related queue managers, WebSphere MQ FTE requires a *coordination queue manager*. The coordination queue manager must be running at WebSphere MQ 7.0, but does not require a WebSphere MQ FTE installation. WebSphere MQ 7.0 is required because WebSphere MQ FTE uses WebSphere MQ publish/subscribe capabilities. Although WebSphere MQ FTE offers a flexible set of architectural options for connecting agents, queue managers, and coordination queue manager, the LGI scenario deploys the coordination queue manager on a back-end AIX queue manager on a dedicated machine. This deployment is mainly because the other queue managers are running at WebSphere MQ 6.0 and are not yet scheduled for migration to WebSphere MQ 7.0.

Figure 1-4 illustrates the WebSphere MQ FTE architecture.

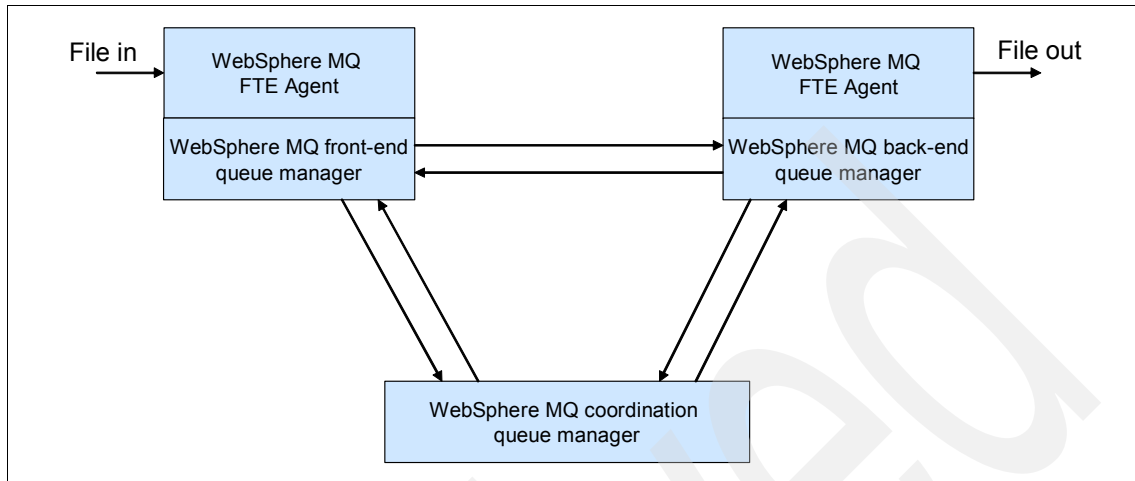


Figure 1-4 WebSphere MQ FTE architecture

Figure 1-4 illustrates the following concepts:

- ▶ The WebSphere MQ FTE agents connect to their respective local queue managers by using bindings connections.
- ▶ All three queue managers are connected to one another in the WebSphere MQ cluster and are secured with a Secure Sockets Layer (SSL).

In addition, file transfer is direct from the front-end queue manager to the back-end queue manager and does not occur by using the coordination queue manager. The coordination queue manager routes administrative commands and status information and does not act as a routing point for file transfer.

Monitoring with IBM Tivoli Composite Application Manager

Software monitoring is required to detect proactively any runtime problems with hardware and software. Without such monitoring, problems only become evident when reported by users in the event of a failure. The ability to collect historical data for trend and bounds analysis is an additional requirement.

Tivoli provides a broad portfolio of monitoring products and offerings. One such offering, IBM Tivoli Composite Application Manager for SOA Platform, provides monitoring facilities for the key products deployed in the LGI scenario. IBM Tivoli Composite Application Manager for SOA Platform is designed for architects, subject matter experts, administrators, and operators who need to deploy SOA applications based on an ESB and want an end-to-end view of their complex applications.

IBM Tivoli Composite Application Manager for SOA Platform is a bundle of products that includes the following products:

- ▶ IBM Tivoli Monitoring Base products (for example, IBM Tivoli Enterprise Management Server)
- ▶ IBM Tivoli Composite Application Manager for SOA
- ▶ IBM Tivoli Composite Applications Manager for WebSphere
- ▶ IBM Tivoli OMEGAMON® XE for Messaging

The bundle provides good monitoring capabilities for the LGI Web applications, ESB, and systems.

Although the LGI scenario can benefit from widespread monitoring of the system, products, and applications, at this stage of the project, we installed the IBM Tivoli Monitoring Base product infrastructure and proved that infrastructure by monitoring the front-end Web applications. This provides a good base for future exploitation of a wider set of the IBM Tivoli Composite Application Manager for SOA Platform monitoring capabilities (for example, monitoring of the ESB).

The IBM Tivoli Monitoring infrastructure is deployed at the back end on a dedicated machine that contains the following product components:

- ▶ IBM Tivoli Enterprise Management Server
- ▶ IBM Tivoli Enterprise Portal Server
- ▶ IBM Tivoli Enterprise Portal

An IBM Tivoli Enterprise Management Agent and an IBM Tivoli Composite Application Manager for WebSphere Data Collector are deployed on the front-end machines.

The deployment provides monitoring of front-end Web applications that we view by using the Tivoli Enterprise Portal.

Choice of front-end application server

The LGI scenario uses front-end application servers running on both WebSphere Application Server and WebSphere Portal Server. Both front-end application servers provide the same functionality to the user. The two types of front-end application servers are tested in this scenario to provide test coverage of the integration points between multiple products.

In a real-world deployment scenario, it is unusual to provide identical functionality using the different products. WebSphere Application Server provides a reliable, highly available, secure, and scalable environment on which to run applications. WebSphere Portal Server provides these features and functionality and adds the ability to build flexible solutions through the use of composite applications.

These composite applications can be customized to the users' requirements based on choices such as the users' role. When building scenarios similar to the one in this book, carefully consider the features of the two products and choose the most appropriate for your scenario.

CICS/WebSphere MQ communication

Business restrictions required that to reduce costs, no changes were to be made to the existing LGI CICS back-end system (any new client applications to use the existing communications area or COMMAREA interfaces to the CICS programs).

To meet this requirement, the WebSphere MQ-CICS bridge was used to allow the LGI back-end system to interact with the other components in the quote and policy system. The CICS-WebSphere MQ adapter can be used as an alternative to the bridge in circumstances where modifications to the CICS applications are acceptable.

Another alternative that was considered, but not used, is the WebSphere Message Broker CICS Request node. This was an acceptable solution, but required that CICS to be running (compared to the WebSphere MQ, which can queue a message if CICS is temporarily offline).

Web Services Security

The LGI scenario contains Web services that perform key business functions. The data passed between the Web service clients and the Web services contains sensitive information, such as a customer's identity and financial details. This information must be protected to prevent it from being read or altered by an unauthorized party.

WS-Security provides message-level security to secure SOAP messages through the use of the following methods:

- ▶ *Authentication* by using security tokens to validate the client
- ▶ *Integrity* by using an XML digital signature to sign the contents of a SOAP message to ensure that information is not tampered with
- ▶ *Confidentiality* by using XML digital certificates to encrypt the SOAP message to ensure that no unauthorized party can access the information

Specifications for WS-Security are defined by the Organization for the Advancement of Structured Information Standards (OASIS). For details, see the following Web address:

<http://www.oasis-open.org/specs/index.php#wssv1.0>

Message-level security is preferable over transport security in the following situations:

- ▶ The Web service calls flow through one or more intermediaries.
- ▶ The Web service calls flow through multiple different transports.
- ▶ The Web service calls use asynchronous queues (transport security leaves the message unsecured while it is held on a queue).
- ▶ Specific parts of the Web service message require different levels of security.
- ▶ The Web service implements authentication and requires the Web service client to pass an identity token

In the LGI scenario, the Web service calls are point to point, where each Web service has a single Web service client that connects directly to it over SOAP/HTTP. In this way, transport security (configuring SSL on the HTTP channel) is sufficient to provide the level of security that is required. However, to extend the SOA security information, as covered in this document, and to demonstrate and test the steps required to configure WS-Security for Web services, the following methods were implemented in the LGI scenario:

- ▶ *Integrity* (signing) and *confidentiality* (encryption) were configured for the message body (and security token if one exists) of the SOAP/HTTP Web service requests and responses.
- ▶ *Basic authentication* were configured by passing a user name token on the request.

User name tokens for authentication: WebSphere Application Server 6.1 has no support for Password Digest when using user name tokens for authentication. Since the user name and password flow in clear text, the user name token also must be encrypted or the Web service transport secured to prevent the password information from being exposed.

1.6 Security configuration

Security was a major focus of this project. As a result, we explored how features in the platforms and products can be used to provide security for the scenario. This project considered the following “four As” of security and identified how each “A” applied to the scenario:

| | |
|-----------------------|--|
| Authentication | Used to verify that the user is who the user claims to be. For example, the user might be authenticated through use of a user ID and password. |
|-----------------------|--|

| | |
|-----------------------|--|
| Authorization | Used to determine which actions an individual user or group of users can perform. For example, the user might be authorized to read or modify files. |
| Administration | Refers to the tools and methods used to modify the security settings for a particular component or system. |
| Audit | Used to ensure that security policy compliance is maintained and that appropriate notifications are made when potential breaches are detected. |

In addition to these four As, the project addressed how important customer data can be protected while that data is processed by each of the components of the scenario.

The following aspects of security were considered for this project:

- ▶ Enterprise Identity Mapping (EIM)
- ▶ Integrated Cryptographic Service Facility (ICSF)
- ▶ Internet Protocol security (IPsec)
- ▶ Network Authentication Service (Kerberos)
- ▶ Lightweight Directory Access Protocol (LDAP)
- ▶ Public Key Infrastructure (PKI) Services
- ▶ Resource Access Control Facility (RACF®)
- ▶ Secure Sockets Layer (SSL)
- ▶ Software product-level security
- ▶ WS-Security

Components on z/OS use central RACF for directory, authentication, and authorization services. Components on AIX use central IBM Tivoli Directory Server for directory and authentication services. Authorization is done in each component.

Figure 1-5 shows an overview of the security for the AIX, Windows, and z/OS deployment used in the project.

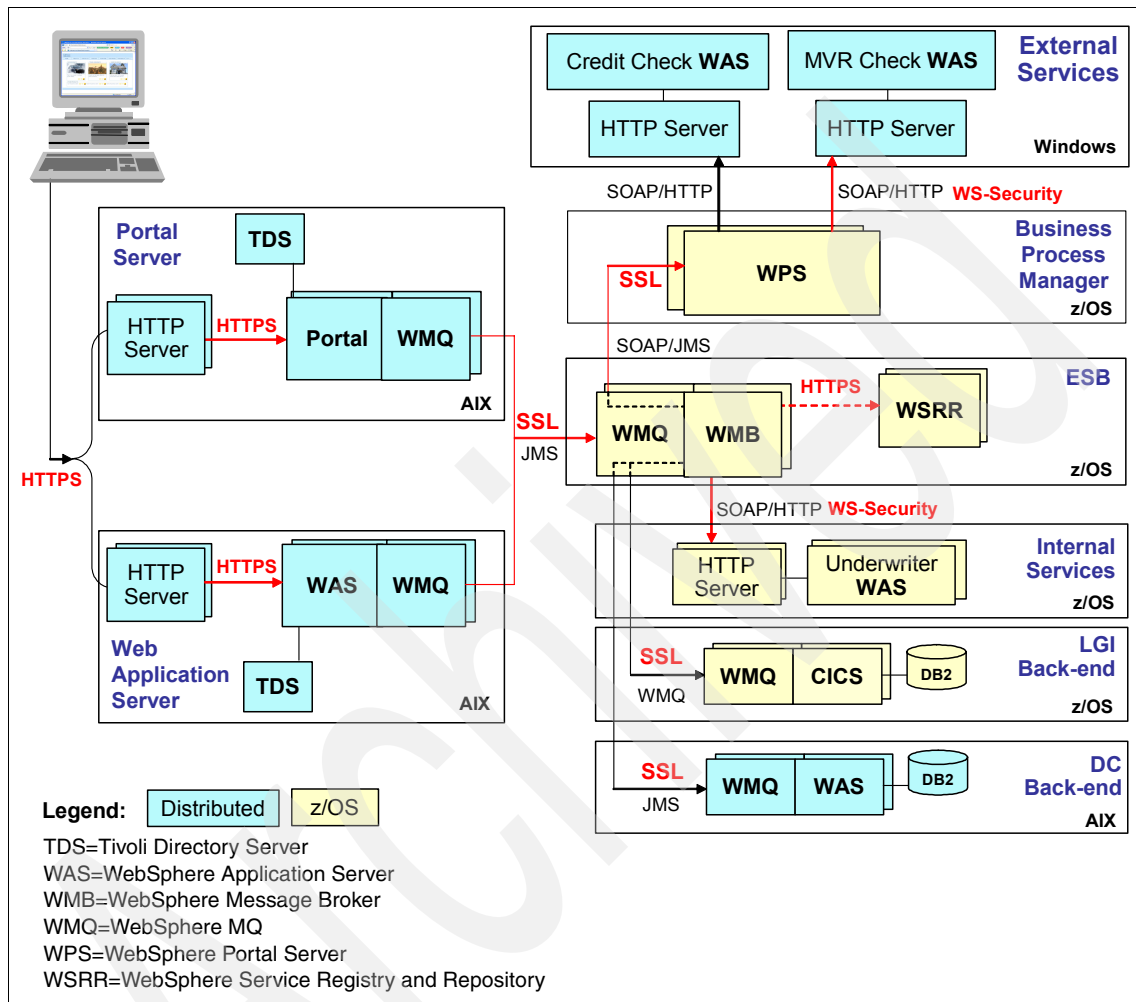


Figure 1-5 AIX, Windows and z/OS transport and message level security overview

Figure 1-6 provides an overview of the security for the z/OS and Linux for System z deployment used in the project.

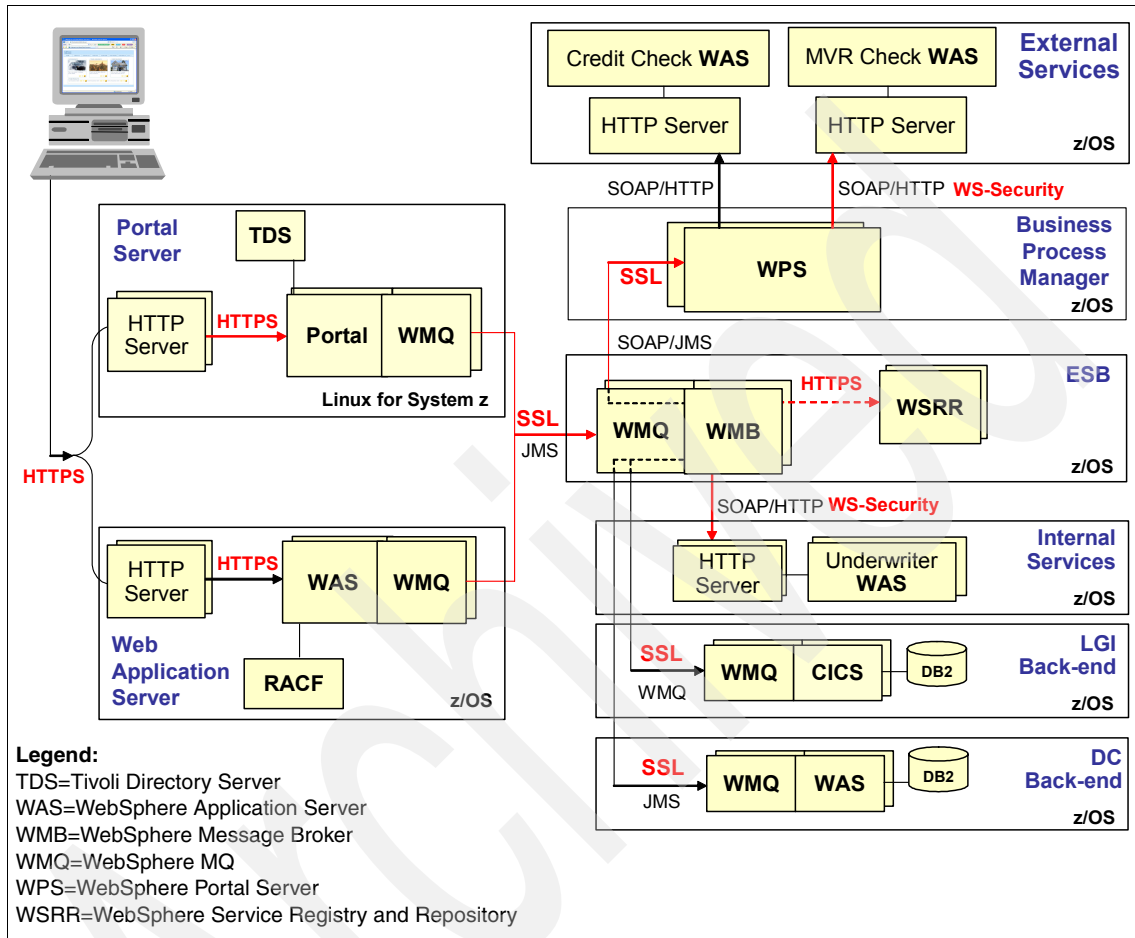


Figure 1-6 z/OS and Linux for System z transport and message level security overview

Figure 1-7 provides a visual overview of the security implemented for the LGI scenario's front-end components in the Mixed Platform Stack (AIX, Windows and z/OS) environment:

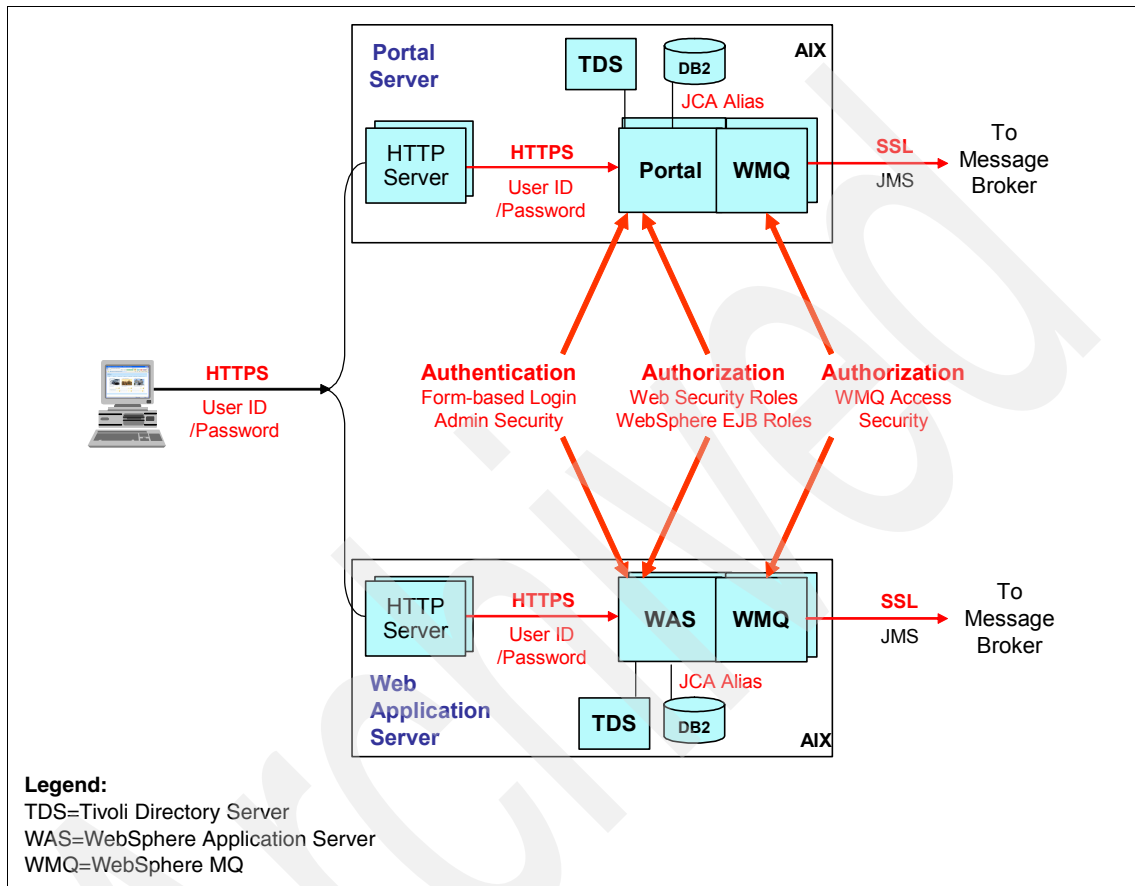


Figure 1-7 Security implemented in WebSphere Application Server and WebSphere Portal Server on AIX

Figure 1-8 provides an overview of the security implemented for the LGI scenario's front-end components in the z/OS and Linux for System z environment.

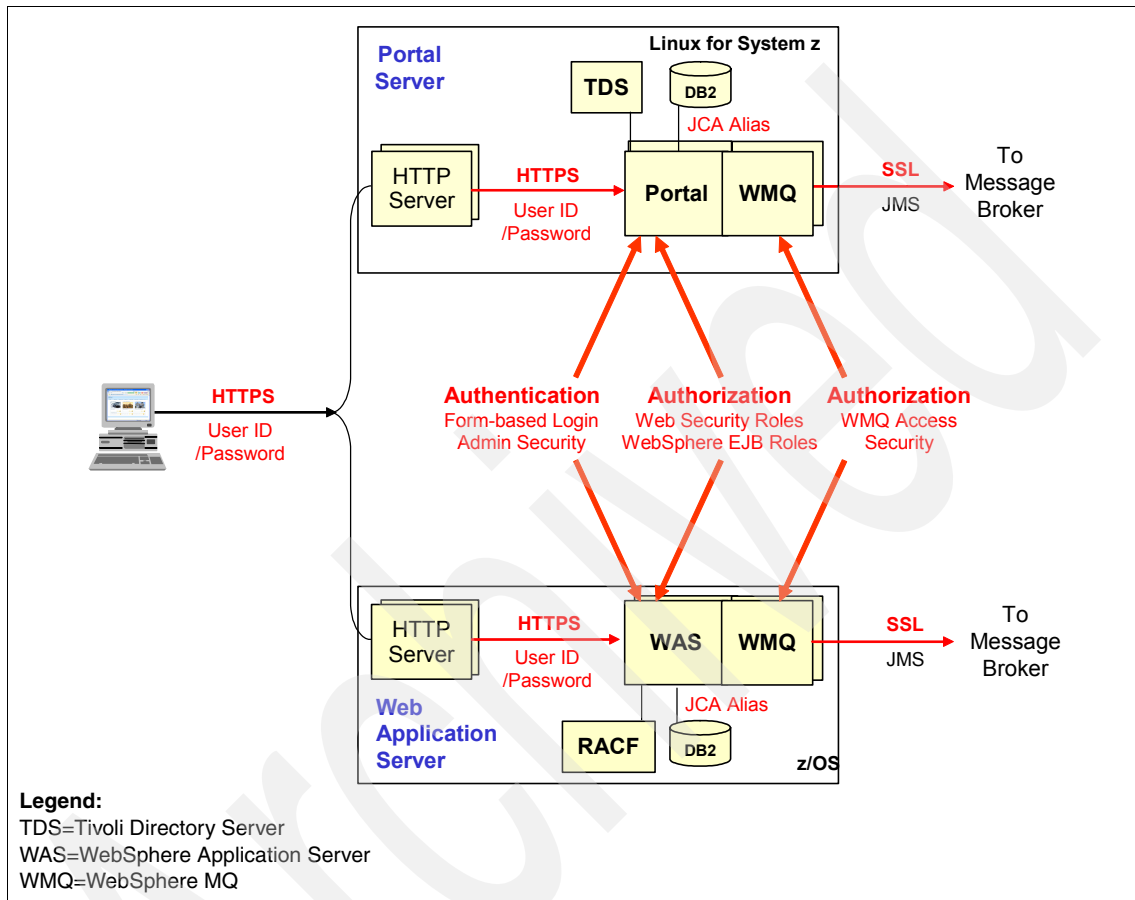


Figure 1-8 Security implemented in WebSphere Application Server and WebSphere Portal Server on Linux for System z

For each product used in the scenario, access to administrative functions is limited to users and groups who require that level of access. User authentication and authorization are used to ensure that users are known and are able to access only those resources for which they have access permission. We test changes to the security settings to ensure that those changes result in the desired behaviors.

The techniques presented in this document can be used to enhance the security of an SOA deployment. They should not be viewed as a complete set of security

features that ensure a deployment is secure. Security policy should be created considering a range of issues, including the following issues:

- ▶ Laws and regulations
- ▶ Service level agreements
- ▶ Company policy
- ▶ Standards
- ▶ Best practices
- ▶ System and product features
- ▶ Knowledge of all data and how data are transported
- ▶ Knowledge of all systems and how systems are connected
- ▶ Potential external and internal users and threats
- ▶ Knowledge of existing and emerging security attacks

It might be necessary to combine the techniques presented in this document with further security enhancements to secure your own deployment. You must carefully analyze your own requirements to ensure that the level of security you implement is appropriate for your deployment.

For further information about SOA and security on System z, see *Designing for Solution-Based Security on z/OS*, SG24-7344.

Archived

Application development

This chapter introduces the application components that we used in the project scenario. It contains an overview of the tools that were used to develop and test the applications. It includes the following topics:

- ▶ “Application overview” on page 28
- ▶ “Development environment” on page 39
- ▶ “Securing the application” on page 42
- ▶ “Securing the development environment” on page 45
- ▶ “Application testing” on page 47

2.1 Application overview

This section describes the flows and components included in the Lord General Insurance (LGI) application scenario.

2.1.1 LGI application flows

The application in the LGI scenario allows new or existing customers to obtain a motor insurance quote and an insurance policy. In addition, customers can either store or accept that quote. If the customer accepts the quote, the application issues a new insurance policy and updates the appropriate back-end system with the customer and policy information. Registered users can retrieve the details of their stored quotes and any existing policies.

The LGI application consists of the following flows:

- ▶ Retrieving existing customer information
- ▶ Getting a quote
- ▶ Accepting a quote

For all task flows, customers interact with the Web application by using a browser-based user interface.

Retrieving existing customer and policy information

By using the Web application, registered users (including customers and LGI employees) can log on to the application. Authentication is performed by using an external security manager such as Tivoli Directory Server or Resource Access Control Facility (RACF). After users log on to the application, they can retrieve from a local database the name and address information along with any stored quotes and existing policies. This application is discussed in more detail in the following sections.

Getting a quote

When the customer uses the Web application to obtain a motor insurance quote, the following actions occur as illustrated in Figure 2-1 on page 29:

1. The Web application sends a Java Message Service (JMS) message to the message broker.
2. The message broker uses WebSphere Service Registry and Repository (WSRR) to discover the WebSphere MQ endpoint queues for the Direct Car (DC) and LGI back-end servers and to discover the SOAP address URL for the underwriter Web service.

3. The message broker sends three concurrent requests:
 - To the DC back-end server to obtain a quote
 - To the LGI back-end server to obtain a quote
 - To the Underwriter Web service to obtain an adjustment value based on the customer's information
4. The message broker aggregates the replies from the DC and LGI back-end servers, selects the lowest quote, and then applies the underwriter's adjustment to the result.
5. The message broker returns the adjusted quote value to the Web application as a JMS message.
6. The Web application presents a message in the user interface to inform the customer of the quote that has been issued. If the quote is for a registered user, the user is offered the option of storing the quote, in which case the Web application writes the information to a local application database.

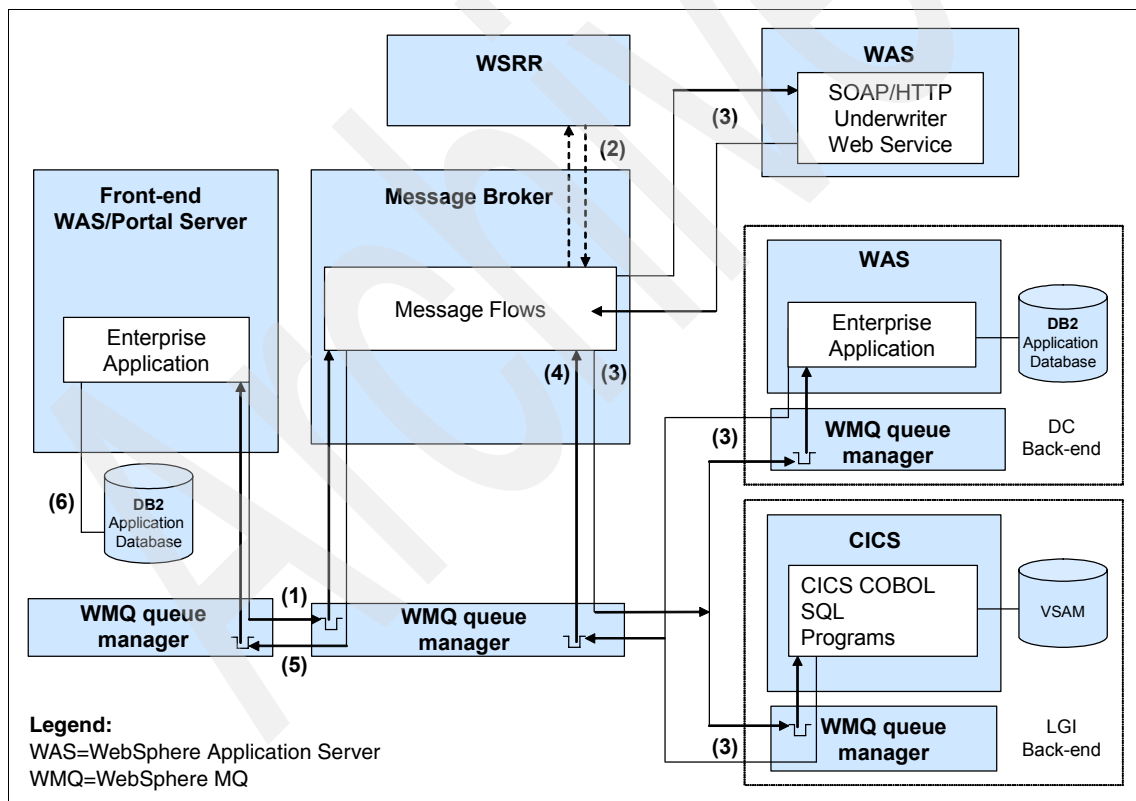


Figure 2-1 Application flow for getting a quote

Accepting a quote

The application flow for accepting a quote involves two parts:

- ▶ A synchronous flow in which the customer accepts a quote and obtains a policy
- ▶ An asynchronous call to perform background checks on the customer and vehicle to be insured

For the first flow, the customer uses the Web application to accept an offered quote and obtain a motor insurance policy. During this process, the following actions occur:

1. The Web application sends a JMS message to the message broker.
2. The message broker uses WSRR to discover the WebSphere MQ endpoint queues for the back-end system (either DC or LGI) that issued the quote.
3. The message broker sends a request to the back-end system (DC or LGI) that issued the quote. The back-end system adds the customer and policy details to its database and returns a policy number to the message broker.
4. The message broker returns the issued policy number as a JMS message to the Web application.
5. The Web application presents a message in the user interface to inform the customer that the policy has been provisionally accepted. If the policy is for a registered user, the Web application writes the policy number to a local application database so that the customer can easily retrieve the information.

Figure 2-2 illustrates the steps for the first part of the application flow for accepting a quote.

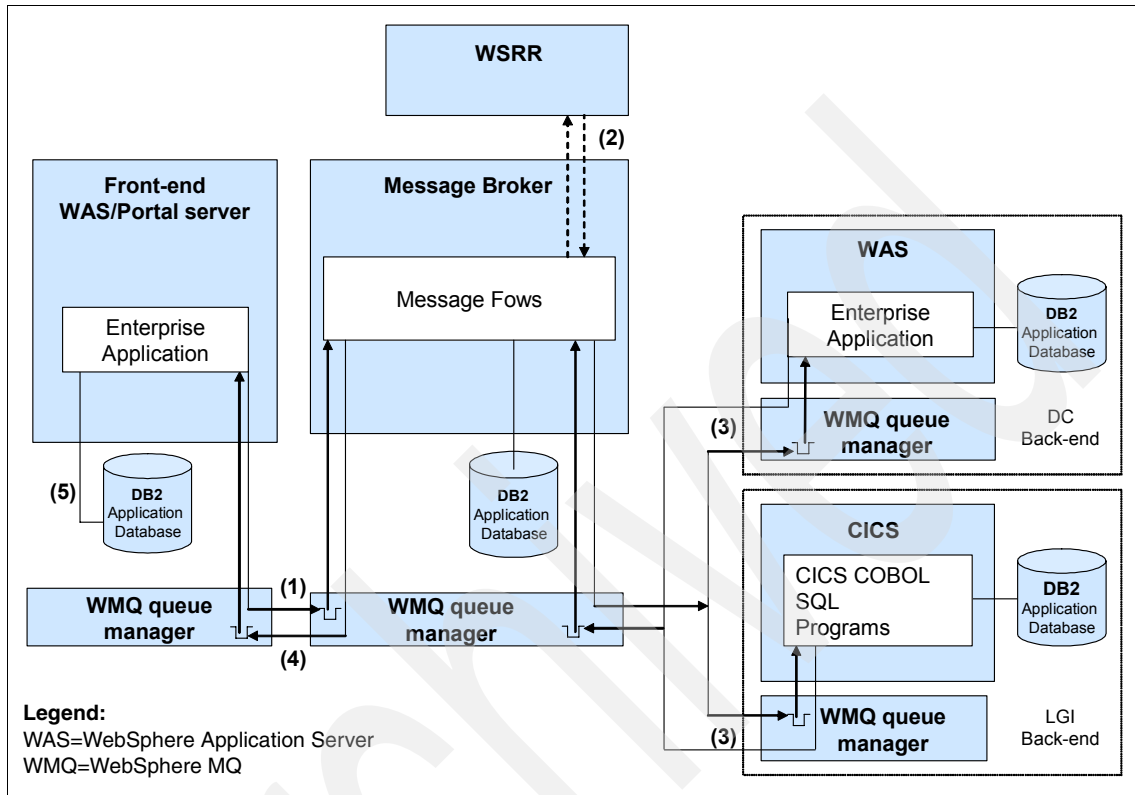


Figure 2-2 Application flow for accepting a quote, Part 1 - Synchronous

The second part of the application flow for accepting a quote involves an asynchronous call to perform the background checks on the customer and vehicle. These checks are required to confirm that the policy can be accepted. During this flow, the following actions occur:

1. At the same time the message broker is returning the policy number to the Web application as described in part one of this flow, the message broker initiates the business process by sending a SOAP/JMS message to invoke a Business Process Execution Language (BPEL) flow in WebSphere Process Server.
2. The BPEL flow makes two SOAP/HTTP requests to call the Credit Check and Market Value Reduction (MVR) Check Web services to perform the necessary checks.

3. The BPEL flow aggregates the results from the two Web services and returns the consolidated result to the message broker.
4. The message broker stores the result in its application database. In a real-world environment, this action triggers notification to customers that their policies have been formally accepted.

Figure 2-3 illustrates the steps for the second part of the application flow for accepting a quote.

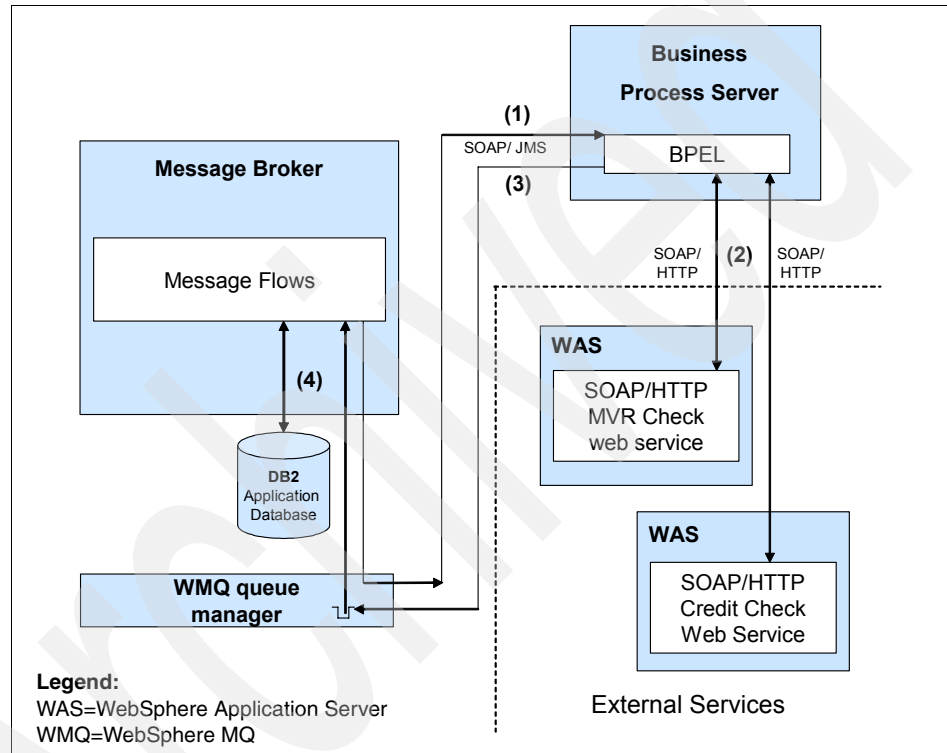


Figure 2-3 Application for accepting a quote, Part 2 - Asynchronous

2.1.2 LGI application components

The LGI scenario consists of multiple application components, including enterprise applications, message flows, and COBOL programs. These application components are deployed to appropriate products within the environment. The key application components include a WebSphere Application Server or Portal front-end server, WebSphere Message Broker, WebSphere Service Registry and Repository, the DC back-end server, the LGI front-end server, WebSphere Process Server, and internal and external Web services.

WebSphere Application Server and Portal front-end servers

For this project, we developed a secure version of the Web application by modifying the user interface of the existing LGI quote application. In doing so, we made a clearer distinction between new and existing (registered) users and the functions that each can perform by using security roles.

There are two alternative implementations of the Web application. Each demonstrates different technologies, but both offer the same functionality to users:

- ▶ JavaServer Pages (JSP) and servlets hosted on WebSphere Application Server
- ▶ Portlets hosted on WebSphere Portal Server

Both invoke Enterprise JavaBeans (EJB; session and entity beans) hosted in WebSphere Application Server to access a DB2 application database and issue JMS API calls.

The front-end server performs the following key functions, which are shown in Figure 2-4 on page 34:

- ▶ Provides a browser-based Web interface for the user (insurance customer)
- ▶ Provides a page where registered users can log in and obtain authentication and authorization to use restricted functions within the application
- ▶ Initiates all requests to back-end systems by using JMS calls to the message brokers.
- ▶ Uses entity beans to inquire on and insert to a DB2 database and stateless session beans to control access to the entity beans

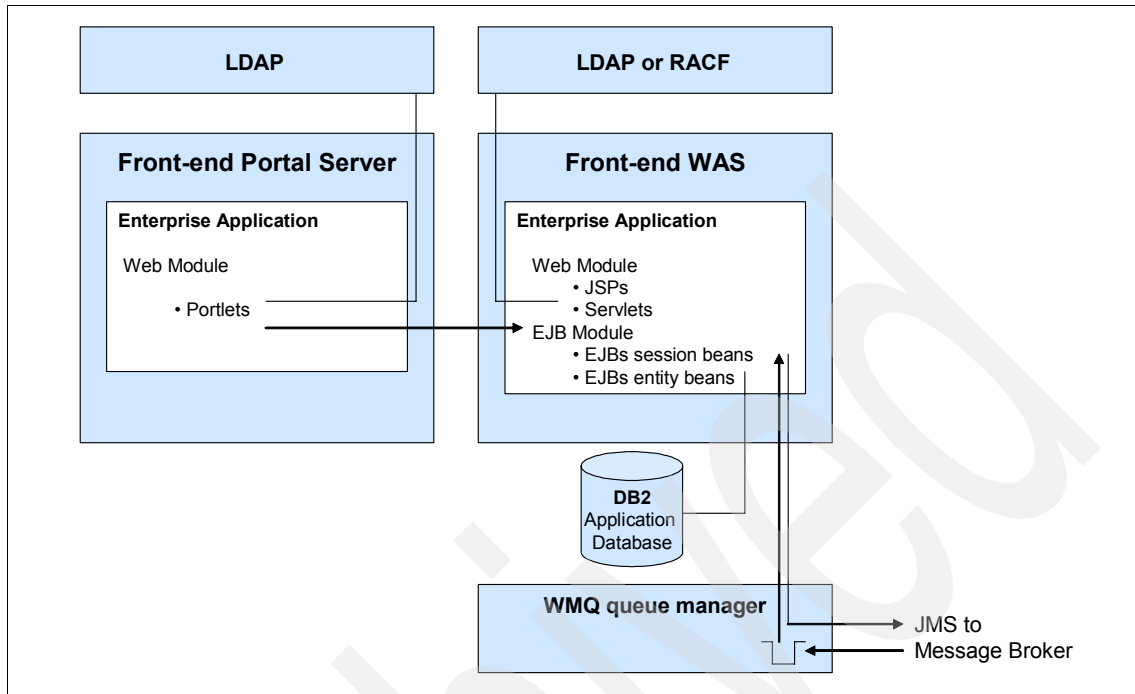


Figure 2-4 Front-end Web application logical design

WebSphere Message Broker

WebSphere Message Broker performs the following key functions, which are shown in Figure 2-6 on page 36:

- ▶ Serves as the LGI scenario ESB, providing a link between the front-end WebSphere Application Server or Portal servers and the business process manager, Web services, and back-end systems
- ▶ Performs transformation between the JMS message received from the WebSphere Application Server front-end server and the following formats:
 - The JMS format required by the DC back-end system
 - The WebSphere MQ format required by the WebSphere MQ-CICS bridge on z/OS (conversion to WebSphere MQ and addition of MQ CIH header) for the LGI back-end system
- ▶ Performs aggregation during the quote message flow, sending requests to both the DC and LGI back-end systems and aggregating the replies
- ▶ Looks up a WebSphere MQ service in a WSRR server (using the endpoint lookup node) and then parses the returned WebSphere MQ Internationalized Resource Identifier (IRI) to invoke the LGI or DC back-end servers

For more information, see SOA messaging in 4.6.3, “SOA messaging flow development” on page 125.

Internationalized Resource Identifier: The IRI is a protocol that complements the Uniform Resource Identifier (URI). IRIs can contain any Unicode characters, where URIs are restricted to ASCII characters. A mapping from IRIs to URIs is defined, which means that IRIs can be used instead of URIs.

- Looks up a Web service in a WSRR server (using the endpoint lookup node) and then uses the returned URL to invoke the underwriter SOAP/HTTP Web service (using the SOAPRequest node)
- Coordinates a two-phase commit transaction (for the Add Customer and Add Policy requests to the LGI back-end server) using the UOWControl flag in the CICS Bridge (MQ CIH) header during the accept quote flow

In the case of the DC back-end system, the message broker sends a single request and the DC WebSphere Application Server server coordinates the transaction.

The CICS node can be used as a more efficient alternative to this approach and applies to WebSphere Message Broker on z/OS only. The approach chosen for this project allows the message flows to run unchanged on multiple platforms.

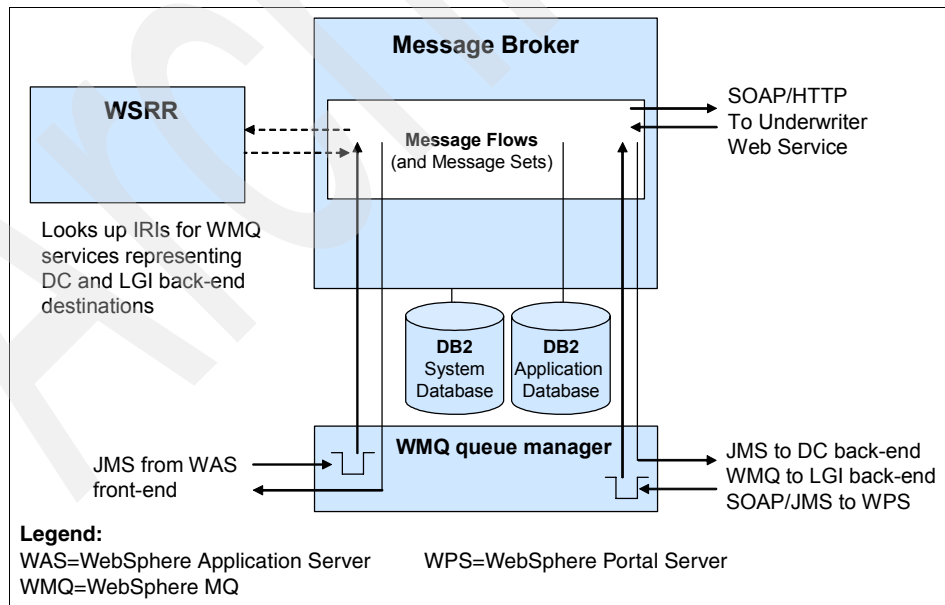


Figure 2-5 WebSphere Message Broker application logical design

DC back-end server

The DC back-end server performs the following key functions, which are shown in Figure 2-6:

- ▶ Is invoked by JMS messages from the Message Brokers, which trigger message-driven beans (MDBs) that in turn invoke EJB session beans
- ▶ Accesses a local DB2 database by using Java beans that make Java Database Connectivity (JDBC™) API calls to inquire and add customer and policy data and obtain quote values

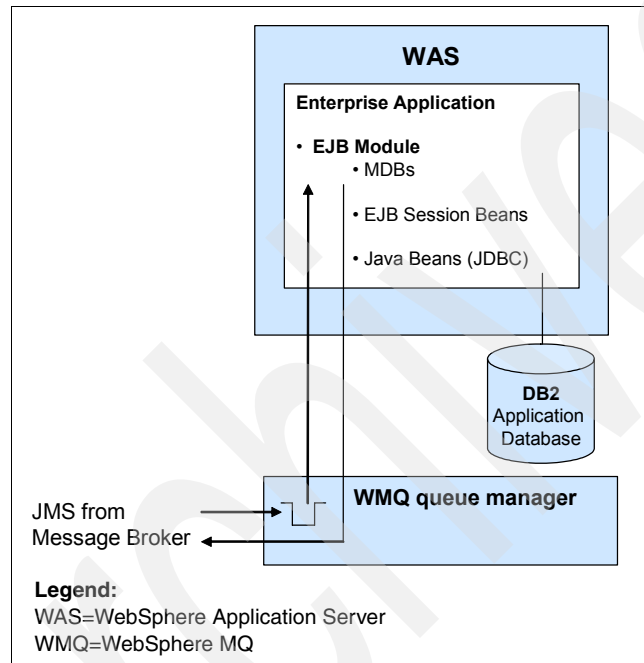


Figure 2-6 DC back-end application logical design

LGI back-end server

The LGI back-end server performs the following key functions, which are illustrated in Figure 2-7:

- ▶ Is invoked by WebSphere MQ messages with MQ CIH headers from the message brokers that trigger the WebSphere MQ-CICS bridge to start the CICS programs.
- ▶ CICS COBOL programs read from a VSAM file to obtain quotes values.
- ▶ CICS COBOL programs uses EXEC SQL statements to access local DB2 database to inquire and add customer and policy data.

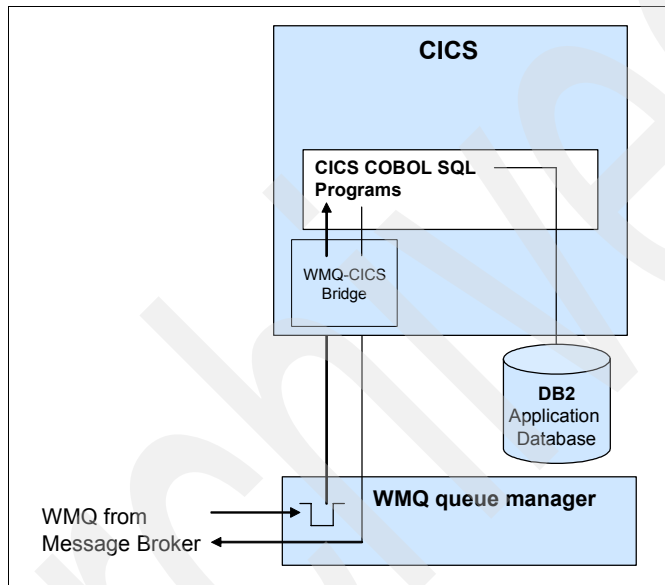


Figure 2-7 LGI back-end application logical design

WebSphere Process Server and external services

WebSphere Process Server and external services perform the following key functions, which are illustrated in Figure 2-8:

- Invokes long running business process (BPEL) as a SOAP/JMS Web service by the message broker

This process runs asynchronously to the rest of the accept policy flow, because it simulates a process that can take several days for completion of all steps.

- Calls two SOAP/HTTP Web services and aggregates the results
- Provides information based on a customer's details through the MVR check and Credit check Web services (simple SOAP/HTTP Web services intended to simulate services performed by another company external to LGI)

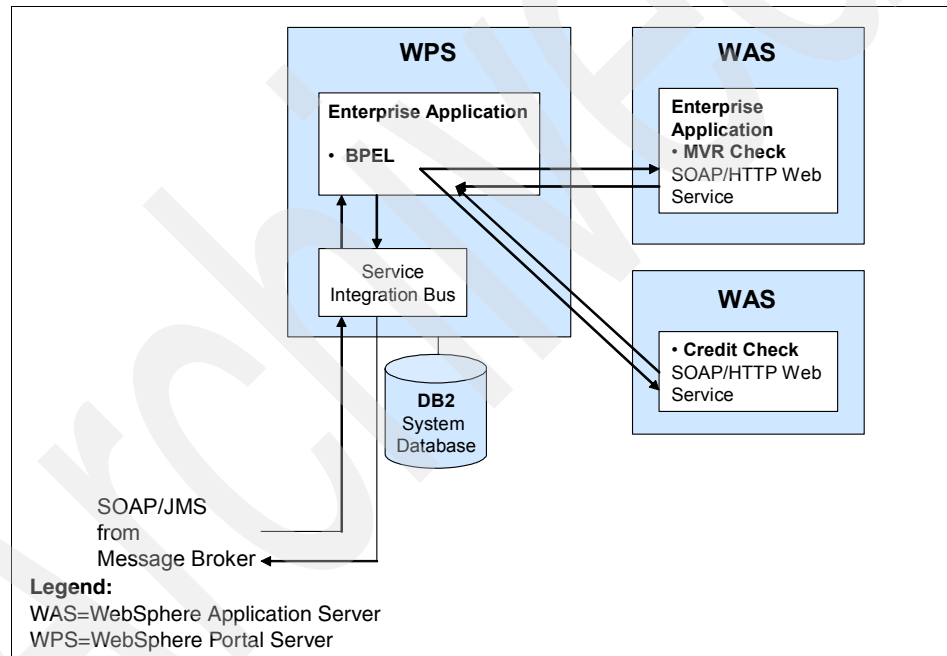


Figure 2-8 LGI WebSphere Process Server and external services application logical design

2.2 Development environment

This section covers development of the secure version of the Web application and the portlets. The Web application contains components that are used by the portlets.

2.2.1 Web application development

We used Rational Application Developer v7.0 on Windows to develop and test the Web application. Rational Application Developer v7.0 is compatible with the target runtime version of WebSphere Application Server 6.1. It provides various levels of WebSphere Application Server and can quickly deploy code in development to these servers for testing. Rational Application Developer helped to reduce development time by allowing immediate testing of changes to the application.

The Web application consists of the following projects:

- ▶ An enterprise application project
- ▶ A dynamic Web project containing JSP and servlets
- ▶ An EJB project containing session and entity beans

The following Rational Application Developer features were used:

- ▶ The Data perspective to connect to a copy of the target database
- ▶ The Data perspective “Create EJBs from Table” wizard to generate the entity beans
- ▶ The J2EE perspective to develop the session beans
- ▶ The Web perspective to generate the servlet and user interface components

Tip to avoid `org.omg.CORBA.portable.UnknownException`

The `org.omg.CORBA.portable.UnknownException` error occurs in a client application if it calls a session bean method which throws a `javax.ejb.EJBException` and that exception is not declared. To avoid this error when invoking EJBs, use the following tips:

- ▶ For each public method in the session bean that throws a `javax.ejb.Exception`, add throws `javax.ejb.EJBException` to the method signature.
- ▶ Ensure that throws `javax.ejb.Exception` is also reflected in the session bean's local and remote interface method signatures. If the method has already been promoted to the local interface, the remote interface, or both, in Rational Application Developer, regenerate the method signature definition in the appropriate interfaces:

- a. Open the session bean code in the J2EE Perspective.
- b. Right-click the methods in the Outline view, select **Enterprise Bean** and click **Demote from Local Interface** or **Demote from Remote Interface** as appropriate. Save the file, but do not close it.
- c. Right-click the methods in the Outline view, select **Enterprise Bean** and click **Promote from Local Interface** or **Promote from Remote Interface** as appropriate. Save and close the file.

Tips for generating entity beans

When generating entity beans from DB2 databases, ensure that the source database does not use DB2 reserved words for any column names. Failure to do this can cause runtime errors because the method that the WebSphere Application Server uses to generate the runtime entity bean logic.

For further details about this topic, see the following sections in the WebSphere Application Server 6.1 Information Center:

- ▶ *SQL reserved keywords for DB2 Universal Database for z/OS V9*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.etools.ejbbatchdeploy.doc/topics/rsqldb2uDBOS390_V9.html
- ▶ *SQL reserved keywords for DB2 Universal Database V9.1 for Linux, UNIX, and Windows*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.etools.ejbbatchdeploy.doc/topics/rsqldb2uDB_V91.html

Tip for viewing SQL statements in generated code

Application development tools, such as Rational Application Developer, provide a user friendly interface to develop entity beans. However, there are times (during problem determination, for example) that it is useful to see the actual SQL statements that have been generated to implement the functions.

Use the following method to view the SQL statements in the generated back-end ID code:

1. Obtain the enterprise application EAR file used in the runtime environment:
 - a. Select **Applications** from the WebSphere Application Server 6.1 Integrated Solutions Console.
 - b. Select **Enterprise Applications**. Then select the check box next to the appropriate application.
 - c. Click **Export**. After the EAR file is displayed, right-click the file name and use the browser's save function to save the file to a local temporary directory.

2. Use an file extraction utility to open the EAR file and then to expand the EJB Java archive (JAR) file within it. (This step might be easier on a Windows PC rather than AIX or z/OS).
3. Sort the contents by path name, locating files with paths of the format *packageName/websphere_deploy/backendID*. There might be files for more than one back-end ID if the option to generate alternative back-end ID code was selected during deployment. (The default back-end ID value for applications developed in Rational Application Developer 7.0 is DB2UDBNT_V82_2).
4. For each entity bean, open the *EntitybeanNameFunctionSet_nnnnnnnn.class* file in the back-end ID path that is used by your runtime environment.

Because it is a compiled class file, much of the information is unreadable. However, the SQL statements are readable. Look for SELECT, INSERT, UPDATE or DELETE statements. This provides enough information to see if an explicit schema name is used and the column names are being operated upon.

2.2.2 Portlet development

We used Rational Application Developer v7.5 on Windows to develop and test the portlet applications. Rational Application Developer v7.5 is compatible with the target runtime version of WebSphere Portal Server v6.1. The Web application components of the LGI scenario were developed prior to the general availability of Rational Application Developer v7.5. Version 7.5 has all the features that are necessary to develop the Web application components and can be used in place of Rational Application Developer v7.0 if desired.

The portlets developed for the LGI scenario are all basic portlets that use the JSR168 specification. The portlets use the state pattern and the service locator pattern to simplify the code. They were developed by following the guidance in the *Portlet Development Workbook*, which you can find at the following address in IBM developerWorks®:

http://www.ibm.com/developerworks/websphere/library/techarticles/0608_hanis/0608_hanis.html

The portlets are packaged as a single Web archive file (WAR file) the contains the following portlets and files:

- ▶ New quotes portlet
- ▶ Your quotes portlet
- ▶ Your policies portlet
- ▶ Your details portlet
- ▶ EJB client JAR file
- ▶ Service locator manager JAR file

The optional portlet and portal tools must be installed in Rational Application Developer to enable you to develop portlets. The Portal Development activity must be enabled to allow portlet development. Rational Application Developer prompts you to enable the appropriate activities the first time either a new Portal project or Portlet project is created. We installed WebSphere Portal Server 6.1 on the same machine as Rational Application Developer v7.5, which allowed us to quickly deploy portlets to a test environment as the portlets were developed.

Tip: The performance of a development and test environment can be optimized by choosing an administration installation and following the instructions in “Enable development tasks” table (Table 1) in the WebSphere Portal Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/index.jsp?topic=/com.ibm.wp.ent.doc/install/inst_opt.html

2.3 Securing the application

The LGI Web application is secured by using role-based security, with roles on both the Web and EJB modules. The Web module also provides an example of customizing application logic for a specific role.

2.3.1 Configuring the secure pages in the Web application

Security constraints are used in J2EE applications to restrict user access to resources. A security constraint defines which HTTP operations are allowed for a set of URLs defined by URL patterns. Security constraints use roles defined in the application to determine which users can or cannot access specific resources that are protected by URL patterns. When the application is deployed, users and groups in the user account repository are given these roles to allow them appropriate levels of access to the application.

The security checks in J2EE Web applications are performed based on the URL that is used in a particular request. To secure an application, URL patterns are used. When a request matches a URL pattern, the security constraint associated with the pattern is applied, and access is restricted as appropriate.

When an attempt is made to access a secured resource, the user is prompted to log in to the application. A variety of log in mechanisms are provided including the following types of authentication:

- ▶ Basic authentication
- ▶ Form-based authentication
- ▶ Certificate authentication

The LGI Web application uses form-based authentication to authenticate users. When form-based authentication is used, users are directed to a specific login page to provide their user IDs and passwords. The login page also has an associated error page to provide information in the event of an authentication problem. Both the login page and the error page are specific to the application and are defined in the deployment descriptor.

The LGI Web application uses the `forward()` method on the `RequestDispatcher` to navigate between pages. As a result, the URL that is displayed in the browser remains the same while a user navigates through the application. J2EE applications perform security checks based on URLs. The actual URL of each page is hidden by the `forward()` method. The J2EE security checks do not work correctly based on the URL of a given page, because this is hidden by the servlet.

To allow the security checks to work correctly, a servlet mapping must be created that matches the URL pattern associated with the security constraint. In the LGI Web application, the URL pattern used is `/secure/*`. This means that any requests that contain `/secure/` in the URL are restricted. To enable the security checks to work as desired, the servlet mapping `/secure/LGIServlet` is created. This matches the URL pattern and ensures that any request that uses this URL will be subject to the security constraint.

To configure the Web application so that both the secure and non-secure URLs invoke the same servlet (for example, `LGISecureServlet`):

1. Open the servlet configuration properties of the Web module.
2. Add URL mapping for secure URL.

To configure the log in information for the Web application:

1. Open the log in configuration properties of the Web module.
2. In the Log in pane, configure the Web application to use form-based login and set the JSP values.

For step-by-step instructions, see “Instructions for 2.3, ‘Securing the application’” on page 154.

2.3.2 Enabling role-based security in a Web application

In this section, two security roles are defined to a dynamic Web project for use in the LGI application. The following steps restrict the use of Web pages that are associated with the secure URL to one of the roles:

1. Open the security configuration properties of the Web module.
2. Define the roles used by the LGI application Web module.
3. Restrict the HTTP methods for the secure URL.
4. Define the roles that can access the restricted HTTP methods.
5. Configure the client server connection to secure the data content while it is in transit using Secure Sockets Layer (SSL; content includes user IDs and passwords).

To add a security role to an EJB project for use in the LGI application and restrict selected bean methods to that role:

1. Open the assembly configuration properties of the EJB module.
2. Restrict the selected methods to the security role.

For step-by-step instructions, see “Instructions for 2.3, ‘Securing the application’” on page 154.

2.3.3 Customizing logic in the J2EE application using role-based security

To customize application logic for a specific role:

1. Ensure that the security role (for example, *StaffDiscount*) has been defined to the enterprise application by using the **Security** tab of the dynamic Web project.
2. Open the appropriate source code in the Web project (for example, a servlet) and use the following conditional code to restrict logic to a specific role (for example: *StaffDiscount*, where *StaffDiscount* must exactly match the value defined for the role in the deployment descriptor):

```
if (request.isUserInRole("StaffDiscount"))  
{ do specific role based logic }
```

2.3.4 Portlet security

The portlets used for the LGI scenario require no specific application code to meet the security requirements of the scenario. Authentication and authorization are handled by WebSphere Portal Server. The appropriate authorization rules are configured by using the facilities that are provided in the deployment environment.

For additional information about this configuration, see 3.3, “WebSphere Portal Server 6.1 on AIX and Linux for System z” on page 55.

2.4 Securing the development environment

To configure security during application development:

1. Configure administration and application security on the WebSphere Application Server 6.1 test environment on Rational Application Developer. First use local operating system security and then use Tivoli Directory server as an external security manager.
2. Map runtime users and groups to application security roles in the application server that is built in to Rational Application Developer.

The following sections explain these steps in greater detail.

2.4.1 Enabling administration security using the local operating system

Use the following steps to enable administration security in the WebSphere Application Server 6.1 test environment on Rational Application Developer. The local operating system is used for the user account repository.

1. Open the Integrated Solutions Console for the application server.
2. Configure the application server to use local operating security.
3. From the Select Secure administration, application and infrastructure page, configure administration and application security.

For step-by-step instructions, see “Instructions for 2.4, ‘Securing the development environment’” on page 157.

2.4.2 Administration security using Tivoli Directory Server

It is possible to enable administration security in the WebSphere test environment by using Tivoli Directory Server. For instructions, see Chapter 3, “Front-end deployment (AIX, Linux for System z, and z/OS)” on page 51.

2.4.3 Mapping runtime users or groups to application security roles

To map users or groups from the available security realm in the application server to the application security roles used in the LGI application:

1. From the Integrated Solutions Console, select **Applications** and then **Enterprise Applications**.
2. Select the LGI Web application **LGDSecureQuoteEAR**.
3. Select **Security role to user/group mapping**.
4. Select **Staff Discount Role** and select **Look up groups**.
5. Use the search function to navigate to the group that represents the LGI employees. Ensure that it is displayed in the Selected column, and click **OK** to restrict this role to users of this group.
6. For Roles, select **All authenticated?** for “All Quote Users and EJB Quote Users” to allow all users that are logged on to perform these roles.
7. Click **OK**.
8. Restart the application for the changes to take effect

2.4.4 Troubleshooting common problems with securing the server

You might experience the following problems when securing the WebSphere test environment:

- **Problem:** After enabling administrative security, the status of the server displays incorrectly in the workbench.

Solution: Rational Application Developer presents a message that sends you to a Technote, at the following address:

http://www-1.ibm.com/support/docview.wss?rs=2042&context=SSRTLW&context=SSJM4G&context=SSSTY3&context=SSCGQ7C&q1=server+status+starting&uid=swg21266028&loc=en_US&cs=utf-8&lang=en

Follow the information at this location to resolve the issue.

- **Problem:** Error ADMA017EE is displayed when associating a user with an application (in Security role to user/group mapping).

Solution: Visit the Technote at the following Web address and follow the information as provided

<http://www-1.ibm.com/support/docview.wss?uid=swg21260016>

This resolves error ADMA017EE, but an additional step is required to get the role mapping to take effect. To ensure that role mapping functions properly:

- a. In the Server view, double-click **WebSphere Application Server v6.1** to open the server's configuration file.
- b. In Publishing, select **Run server with resources on Server**.
- c. Save and exit the server's configuration file.

2.4.5 Rational Application Developer V7.5 and WebSphere Portal Server V6.1

An installation of WebSphere Portal Server v6.1 configured for development and test automatically uses a file-based user registry. Initially this contains a single user ID. The user ID is that of the portal administrator and was defined when the product was installed. This user registry is adequate for development and test purposes. Therefore, it is not necessary to configure an alternative registry.

The server is defined to Rational Application Developer by using the New Server wizard. Rational Application Developer correctly detects the administrator user ID for WebSphere Portal and populates the user ID in the New Server wizard.

Be sure to enter the current password for the administrator where prompted, thereby ensuring that the correct password is used. If the correct user ID is not detected, an alternative can be specified. If additional users are required for testing purposes, they can be defined using the Users and Groups portlet. No further security configuration is necessary to accomplish the development and test activities for the LGI scenario.

2.5 Application testing

This section provides information about the testing that the team performed for the newly-developed secure Web application. This testing included the session and entity beans and the security roles. (This version of the application introduced a database to store registered users' information.)

We did not conduct explicit testing of JMS calls in the application development environment. We made this decision because of the fact that the JMS logic in the application was thoroughly tested in an earlier phase of the project. During this test phase, the team commented out the JMS statements and manually entered dummy data to replace the data that is normally received by using JMS replies.

We completed the testing by using the WebSphere test environment. This environment consists of the WebSphere Application Server V6.1 runtime environment integrated into the Rational Application Developer workbench.

2.5.1 Testing the session and entity beans

Rational Application Developer provides a Web application called the *Universal Test Client*. This Web application aids in the testing of session and entity beans. For more information about this Web application, see the “Universal Test Client” section in the WebSphere Application Server 6.1 Information Center, at the following address:

<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.ws.ast.st.common.ui.doc/topics/cwejbt.html>

To test the session and entity beans:

1. Create a local Windows version of the DB2 database that the LGI Web application accesses. Load any required tables with test data.
2. Start WebSphere Application Server V6.1 in the test environment.
3. Log on to the administrative console.
4. From the administrative console, define a JDBC connection to the DB2 database. Use the test connection function to ensure that the application server can access the database.
5. Deploy the Web application to the application server.
6. Use the Universal Test Client to invoke the entity and session beans directly (independently of the servlets and JSP) by following instructions in the “Testing enterprise beans in the universal test client” section in the WebSphere Application Server Information Center, at the following address:

<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.ws.ast.st.common.ui.doc/topics/twejbt.html>

Enterprise bean not displayed as a link: If an enterprise bean is not displayed as a link that you can select in the JNDI Explorer (that is, it is shown as a pale blue diamond) and you have followed the advice in the Information Center, the following steps can help to resolve the problem:

1. In the Universal Test Client, select **WAS v6 Classloading**.
2. If security is enabled on the application server, enter User and Password and select **Set**.
3. Click the **Refresh** icon in the top right corner. The required enterprise application is now displayed under Modules on the classpath.
4. Select **JNDI Explorer** and click the Refresh icon in the top right corner.
5. Navigate to the required JNDI. The enterprise bean is now displayed as a selectable link.

2.5.2 Testing the security roles

To test the security roles:

1. Start the WebSphere Application Server V6.1 server.
2. Log on to the administrative console.
3. Enable administration and application security by using the local operating system in the application server.
4. Deploy the Web application to the application server.
5. Open the Web application URL by using your preferred browser. You can also select on the Web browser icon on the J2EE perspective or the Web perspective toolbar.
6. Run the Web application by using various user IDs and ensure that the application performs as expected.

Front-end deployment (AIX, Linux for System z, and z/OS)

This chapter introduces the front-end components of the Lord General Insurance (LGI) scenario. These components are used to provide the user interface components of the applications and associated security infrastructure.

This chapter includes the following topics:

- ▶ “Front-end overview” on page 52
- ▶ “Tivoli security on AIX and Linux for System z” on page 52
- ▶ “WebSphere Portal Server 6.1 on AIX and Linux for System z” on page 55
- ▶ “WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z” on page 65
- ▶ “WebSphere MQ on AIX and Linux for System z” on page 70
- ▶ “DB2 Enterprise 9 on AIX and Linux for System z” on page 76
- ▶ “IBM HTTP Server 6.1 on AIX” on page 78
- ▶ “WebSphere Application Server 6.1 Network Deployment on z/OS” on page 80
- ▶ “WebSphere MQ on z/OS” on page 88
- ▶ “DB2 Enterprise 9 on z/OS” on page 93
- ▶ “WebSphere MQ File Transfer Edition on AIX” on page 94
- ▶ “IBM Tivoli Composite Application Manager on AIX” on page 96

3.1 Front-end overview

The front-end components of the solution are deployed to AIX, Linux for System z, and z/OS environments. The applications deployed to each environment are identical and are not modified in any way to run on those environments. The z/OS environment uses platform security features to handle the security requirements of the scenario. The AIX and Linux for System z environments use a combination of operating system features and features provided by Tivoli security products to meet the security requirements of the scenario.

In this section, we provide information about the deployment to AIX, Linux for System z, and z/OS.

The following products are deployed on the z/OS front-end server:

- ▶ WebSphere Application Server 6.1
- ▶ WebSphere MQ 6.0
- ▶ WebSphere MQ File Transfer Edition 7.0
- ▶ DB2 9
- ▶ IBM HTTP Server 6.1

The following products are deployed on the AIX and Linux for System z front-end server:

- ▶ WebSphere Portal Server 6.1
- ▶ WebSphere Application Server 6.1
- ▶ WebSphere MQ 6.0
- ▶ WebSphere MQ File Transfer Edition 7.0
- ▶ DB2 9
- ▶ IBM HTTP Server 6.1
- ▶ Tivoli Directory Server 6.1
- ▶ Tivoli Composite Application Manager for WebSphere 6.1 (AIX only)

3.2 Tivoli security on AIX and Linux for System z

We used IBM Tivoli Directory Server to secure the AIX front-end server. IBM Tivoli Directory Server is the IBM implementation of the Lightweight Directory Access Protocol (LDAP). We stored directory information in a DB2 database that we used to verify users who log in to the system.

3.2.1 Installation

Installation instructions are available in the Directory Server Version 6.1 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.IBMDS.doc/welcome.htm>

Prerequisites for installation

Install the following prerequisites:

- ▶ C++ Version 9.0.0.3 (a prerequisite for DB2 Version 9.1)
- ▶ DB2 Enterprise Server Edition Version 9.1.4 or latest Fix Pack

Preparing the operating system

To prepare the operating system for installation:

1. Turn on asynchronous I/O, which is required for a directory server.
2. Create a group and a user ID for the directory server instance.
3. Set a password for the new user. Substitute *pwd* with a password that is appropriate for your environment.
4. Add root as a member of the new group.
5. Ensure that there is enough free space in the file systems. Enter **df -k** to see the current space in the file systems in kilobytes. Expand the file systems to give the required free space.

For step-by-step instructions, see “Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’” on page 160.

Installing the directory server

IBM Tivoli Directory Server is installed by using the typical or custom installation path. A typical installation installs all features, while a custom installation allows the user to select features to install.

Prerequisite products can be installed with the directory server, but installing them first allows more installation flexibility. It is generally advisable to use the latest level of all software.

For step-by-step instructions for installation using the Install Shield graphical user interface, see “Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’” on page 160.

After the Directory Server is installed and to complete the installation and configuration process:

1. Configure the directory server instance.
2. Start the directory server instance.
3. Start the administration daemon.
4. Start the Web application server.
5. Start the Web Administration Tool.
6. Add the directory server to the Web Administration Tool.
7. Create the LDAP structure.

For step-by-step instructions for completing the installation and configuration process, see “Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’” on page 160.

3.2.2 Creating the LDAP structure

The LDAP structure is created by using the Web Administration Tool. Groups and users are defined as required for the scenario that is validating authentication by using LDAP at various times.

The LDAP structure for the LGI system

Figure 3-1 illustrates the LDAP structure that we used in the LGI scenario.

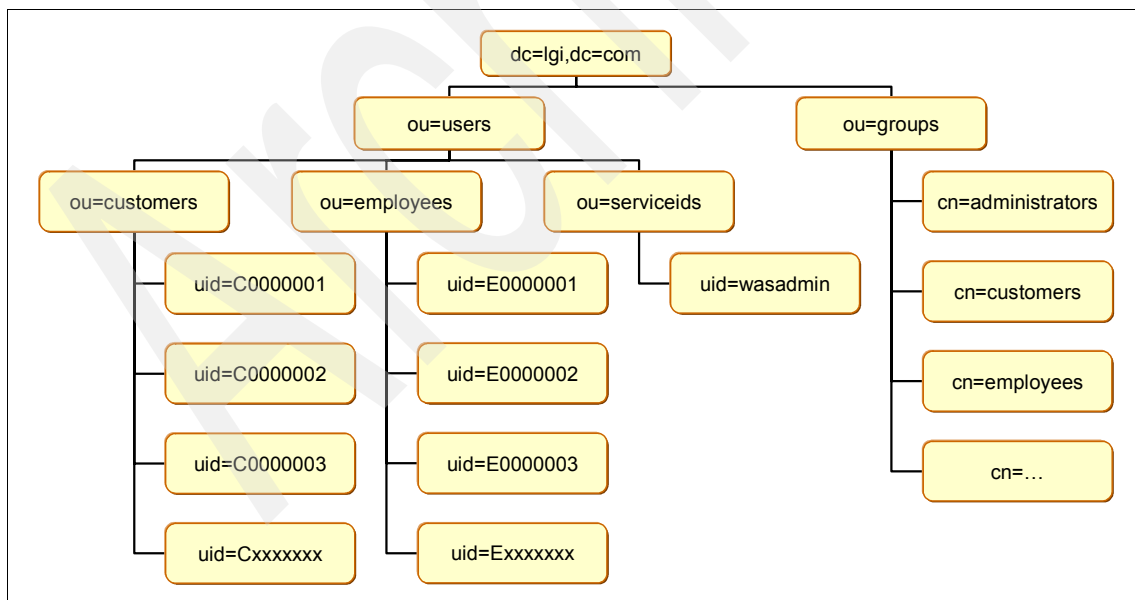


Figure 3-1 LDAP structure for the LGI scenario

Building the LDAP structure

To build the LDAP structure:

1. Log in to the directory server that is added to the Web Administration Tool.
2. Create the suffix.

A suffix (also known as a *naming context*) is a distinguished name (DN) that identifies the top entry in a locally held directory hierarchy. Because of the relative naming scheme used in LDAP, this DN is also the suffix of every other entry within that directory hierarchy. This definition is found in the *Managing suffixes* section of the Tivoli Directory Server Version 6.1 Information Center, at the following address:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDS.doc/install17.htm#mgsuff>

3. Create the domain.
4. Create the groups.
5. Create the users.

For step-by-step instructions for building the LDAP structure, see “Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’” on page 160.

3.3 WebSphere Portal Server 6.1 on AIX and Linux for System z

In this section, we explain how to configure WebSphere Portal Server 6.1 on AIX or Linux for System z to support the LGI scenario. Because WebSphere Portal Server 6.1 works similarly on the two different platforms or any UNIX-based platform, it is installed, configured, and administered in much the same way. Therefore, the information in this section applies to both platforms unless explicitly stated otherwise.

The LGI scenario supports two different user interfaces that can be accessed by users. The WebSphere Portal Server-based user interface uses several portlets to allow users to access the various functions provided in the scenario. The alternative user interface consists of a Web application running on WebSphere Application Server.

The configuration described here uses clustering. *Clusters* are sets of servers that are managed together and participate in workload management. Clusters enable applications to scale beyond the amount of throughput achieved with a single server. They also enable applications to be highly available because requests are automatically routed to the running servers in the event of a failure.

The servers used by the LGI scenario are clustered across different host machines by using the *horizontal scaling* method. A new feature in WebSphere Portal Server 6.1 is the support of dynamic clusters by using WebSphere Extended Deployment. This new feature is not currently exploited in the LGI scenario. The information in this section relates only to static clusters.

3.3.1 Configuration

WebSphere Portal Server is configured in a clustered environment that is managed by a WebSphere Application Server deployment manager. The environment uses at least two WebSphere Portal Server nodes (Portal nodes) to provide high availability. Additional nodes can be added to the environment to improve both scalability and the ability of the environment to handle the failure of a node. The number of nodes required in the environment depends on the workload of the environment and expected response times.

Requests to the Portal nodes are routed by using redundant load balancers and then through Web servers to balance the workload over the nodes. This method ensures high availability of all components between the users' Web browsers and the nodes. This section describes the configuration of the Web servers. Information about load balancers is not included.

Configuration of the WebSphere Portal Server cluster entails the following high-level tasks:

1. Install WebSphere Application Server Network Deployment on the deployment manager node and create a deployment manager profile. A custom installation package is provided with the WebSphere Portal Server media that will install the deployment manager with the necessary fix packs and fixes that are required.
2. Create a deployment manager profile.
3. Install WebSphere Portal Server on the first Portal node (primary node). On a system running a 64-bit kernel, the 64-bit version of the product is installed by default.
4. Configure the primary node to use DB2.
5. Configure the primary node to communicate with the deployment manager.
6. Configure WebSphere Portal Server to use a user registry. Additional information about this task is in 3.3.2, "Security configuration" on page 57.
7. Create a static cluster.
8. Install WebSphere Portal Server on additional nodes.
9. Add the additional nodes to the Portal cluster.

The steps to configure the primary node to use DB2 and to connect additional nodes to DB2 can be performed by using either the configuration tasks or the configuration wizard. The configuration tasks require you to edit the properties files that contain settings that relate to many aspects of the WebSphere Portal Server configuration and not just the current task. The configuration wizard provides an easier approach to performing the configuration because it presents only the configuration properties that are required for a given task. The configuration tasks and configuration wizard can also be used to configure a user repository as discussed in 3.3.2, “Security configuration” on page 57.

You can find the detailed steps to complete the WebSphere Portal Server installation in the WebSphere Portal Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/install/production.html>

For complete information about the use of the wizard, see the “Configuring WebSphere Portal with the configuration wizard” topic in the WebSphere Portal Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/index.jsp?topic=/com.ibm.wp.ent.doc/config/cfg_wizd.html

The Portal nodes communicate with the rest of the components in the scenario by sending and receiving WebSphere MQ messages through Java Database Connectivity (JDBC) database connections. Each node has a local queue manager and an appropriate configuration to allow WebSphere Portal Server to communicate with the queue manager. The resources defined in WebSphere Portal Server include the following types:

- ▶ JDBC providers
- ▶ JDBC type 4 data sources
- ▶ Queue Connection Factories
- ▶ Queues

The configuration of these resources is identical to the configuration required for WebSphere Application Server. For more information about these types of resources, see 3.4, “WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z” on page 65.

3.3.2 Security configuration

WebSphere Portal Server runs on WebSphere Application Server, and there are many similarities between the products. As a result of this, many of the tasks that can be used to secure WebSphere Portal Server are identical to those used to secure WebSphere Application Server. In 3.4.2, “Configuring security” on

page 66, we provide detailed information about the following security-related tasks:

- ▶ Enabling administrative security
- ▶ Enabling application security
- ▶ Enabling Java 2 security
- ▶ Manual node synchronization
- ▶ Using Secure Sockets Layer (SSL) between the Web server and WebSphere Application Server
- ▶ Changing WebSphere Application Server process identity

Several of these tasks are automatically completed when WebSphere Portal Server is configured to use a user repository. See 3.4, “WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z” on page 65, if you want to further understand the impact of these settings or the methods to enable or change them.

The following tasks are automatically completed when WebSphere Portal Server is configured to use a user repository:

- ▶ Enabling administrative security
- ▶ Enabling application security
- ▶ Enabling Java 2 security

The configuration of the user repository is a key area where WebSphere Portal Server differs from WebSphere Application Server. WebSphere Portal Server provides its own tools to configure the user repository. The user repository can be configured by using either the configuration tasks or the configuration wizard as discussed in the steps that follow.

WebSphere Portal Server can use a variety of user repositories for authentication and authorization purposes. The default user repository configuration used by WebSphere Portal Server uses a file based user repository. The default user repository is mainly used in development and test environments. A production environment should be configured to use one of the alternative types of user repository that are more suited to production environments.

The following types of user repository can be used in a production environment:

- ▶ Standalone LDAP user repositories
- ▶ Federated user repositories consisting of one or more LDAP, database, or custom user repositories

The LGI scenario uses a single LDAP user repository that is implemented by using IBM Tivoli Directory Server. In the LGI scenario, the security configuration is performed by using the configuration wizard and choosing to configure a standalone LDAP user repository.

Run the following high-level steps from the primary Portal node to configure security by using the configuration wizard:

1. Define or identify the following users and groups on the LDAP server:
 - One or more portal administrators
 - A portal administrators group that contains the portal administrators
 - A bind user that is used for communication between WebSphere Portal Server and LDAP
2. Optional: Define the following groups if the document and Web content management features of the product will be used:
 - A content administrators group
 - A document reviewers group
 - A Web content management administrators group
3. Start the configuration wizard.
4. Authenticate by using the existing the security settings for WebSphere Portal Server.
5. Select **Configuring Standalone LDAP registry**.
6. Provide appropriate information about the LDAP server and the administrative users defined on the server.
7. Review the information provided and start the configuration.
8. After the configuration is complete, review the resulting message.
9. Verify the correct behavior of WebSphere Portal Server with the new security settings.

Sample LDAP Data Interchange™ Format (LDIF) files to assist with the creation of LDAP users and groups are on the WebSphere Portal Server setup CD or CD image. The information required to complete each panel of the configuration wizard is in the WebSphere Portal Server Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/install/aix_cfg_stndalone_ldap.html

The settings that must be provided to the wizard highly depend on the LDAP structure in use. Additional detailed information about each page can be found by using the help system in the wizard. This help system provides valuable

information about the appropriate values for settings and whether a given setting is optional. The configuration wizard validates the information provided on each panel to ensure correct operation of the configured system. The Portal node must be able to communicate with the target LDAP server for this validation to complete.

The configuration does not occur without successful completion of each validation step. In rare circumstances, it is possible that the configuration step can fail. If this happens, the configuration wizard can be run again after correcting the problem that caused the failure.

Under certain circumstances, it might be necessary to run the configuration wizard to enable security in an environment where global security has already been enabled for the deployment manager cell. This occurs when a standalone Portal node has been federated to a deployment manager where security is already active and a standalone LDAP server is used.

WebSphere Portal Server requires additional security configuration beyond that typically used for a deployment manager environment. The product will fail to start correctly until this configuration is completed. This can be achieved most easily by running the configuration wizard and configuring a standalone LDAP user repository with the same security settings required by deployment manager. In doing so, WebSphere Portal Server completes the additional security configuration that is required for proper operation while maintaining the existing behavior of deployment manager.

As with WebSphere Application Server, if multiple nodes are connected to the deployment manager, the node agent for each node must be running when changes are made to the security settings. If a node agent is not running when security settings are changed, there is a risk that the node will no longer be able to communicate with the deployment manager. This situation is caused by mismatched security credentials. If this situation occurs, it can be rectified by following the steps to manually synchronize the node. When an additional node is added to an existing secure cluster, the new node picks up the security settings for the cell when it is connected to deployment manager.

Since the LGI scenario handles sensitive data, it is necessary to use SSL to encrypt the data transmitted between the users' browsers and the LGI environment. Securing the Web server and the link between the Web server and WebSphere Portal Server was briefly discussed earlier in this section. To complete SSL configuration additional steps that are specific to WebSphere Portal Server are necessary. The steps necessary to complete this configuration are in the WebSphere Portal 6.1 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/security/ssl.html>

3.3.3 Application deployment

The application components of the WebSphere Portal Server front-end server in our scenario consist of four portlets packaged in a single Web archive (WAR) file. The WAR file also contains EJB client code for the various EJB used by the front-end server to communicate with the other components in the scenario. The service locator pattern is used to look up the EJB that are in the environment. The LGI scenario assumes that the portlets and EJB are deployed in the same WebSphere cell, which allows the portlets to look up the EJB resources without requiring additional configuration.

The deployment topology can vary depending on the specific needs for the given scenario. For a small-scale environment, the EJB can be installed on the application servers that comprise the WebSphere Portal Server cluster. In situations where performance and scalability are key concerns, the EJB can be installed on additional instances of WebSphere Application Server in the same cell.

This section covers only the deployment of the portlet applications. Information relating to the EJB deployment is covered in 3.4.3, “Deploying the application” on page 69.

The LGI scenario includes the following portlets:

- ▶ New Quotes Portlet
- ▶ Your Quotes Portlet
- ▶ Your Policies Portlet
- ▶ Your Details Portlet

These portlets are all packaged in a single WAR file. Deploying this WAR file deploys all resources necessary for the portlets to run.

There are three ways to deploy the portlets:

- ▶ Portal administration portlets
- ▶ The XML configuration interface
- ▶ Portal scripting interface

For the LGI scenario, the administration portlets were used to deploy the portlets. To access the administration portlets, log into the portal as an administrative user and click the **Administration** link.

New portlets are installed or updated by using the Web Modules portlet. This portlet allows a WAR file that contains portlets to be uploaded to WebSphere Portal Server and automatically deployed to the Portal nodes in the environment. After the portlets are deployed, they must be activated before they can be used. Attempting to access a portlet before it has been activated leads to an error

message. The Web module should not be activated until it has been synchronized to all Portal nodes in the environment.

The status of the node synchronization can be verified by using the deployment manager administration console. If the nodes are not synchronized, synchronize the nodes prior to activating the Web module. If the Web module is activated prior to this, unpredictable behavior can occur.

The remaining methods for portlet deployment are not covered in this document. For further information about the administration tools, see the “Portal administration tools” topic in the WebSphere Portal Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/admin/admtools.html>

After the portlets are successfully deployed, appropriate access control settings must be applied to grant users the ability to use the portlets. WebSphere Portal Server uses role-based authorization. Roles grant a set of permissions on a restricted resource. A user or group is mapped to a specific role that grants them the necessary level of access to a particular resource.

Resources are organized in a hierarchy that allows the propagation of access control configurations from a resource to its child resources. The child resources inherit access control configurations from their parent. In certain circumstances, it might be undesirable for propagation and inheritance to occur. In such situations, role blocks can be used to prevent either the inheritance or propagation of access control configurations between resources.

Figure 3-2 illustrates the resource hierarchy.

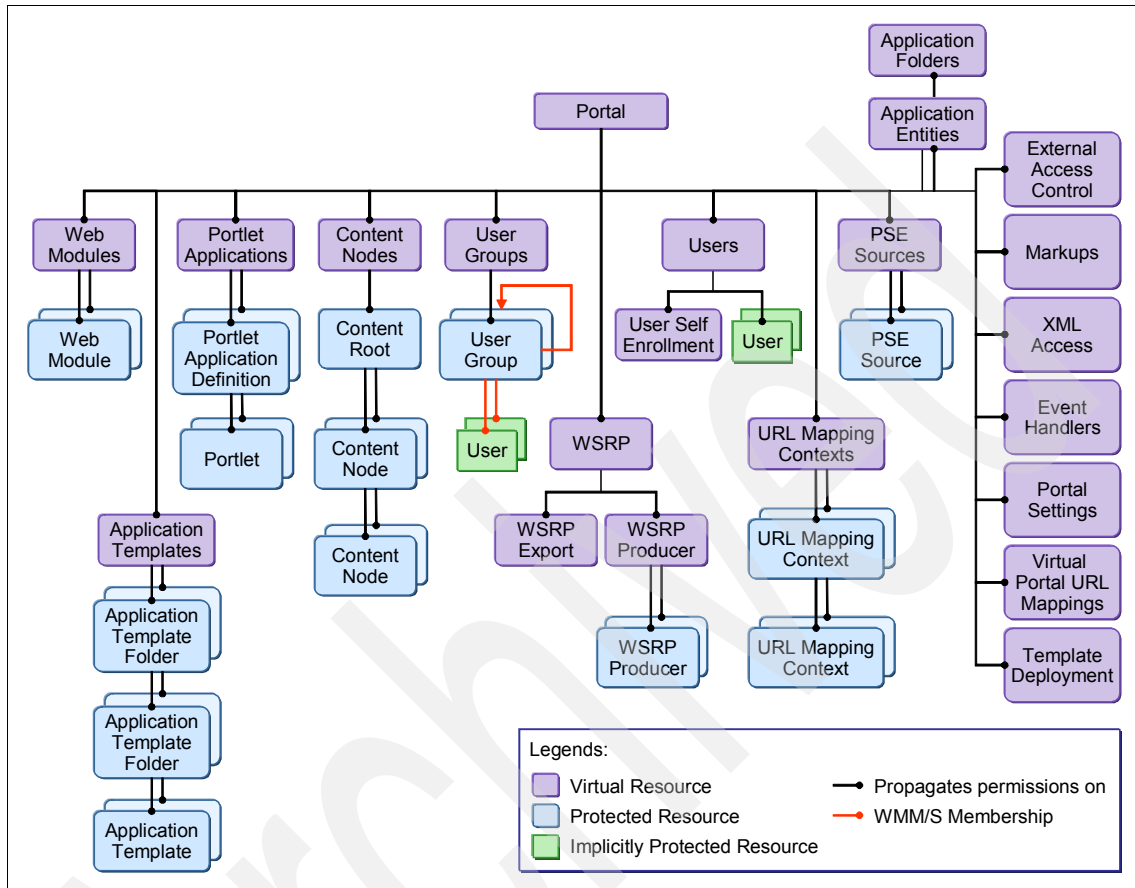


Figure 3-2 WebSphere Portal Server resource hierarchy

Further information about the resource hierarchy and role blocks, see the “Resources” topic in the WebSphere Portal Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/security/sec_resources.html

Roles are also defined in a hierarchy. There are several different roles that can be granted to each resource. A given role grants a specific level of access to a resource plus the level of access of those roles directly beneath it in the hierarchy. For example, the User role allows sufficient access to view a resource. The Administrator role allows full control of the resource. If a given user or group

has not been granted any role on a resource, then they have no access to that resource. Figure 3-3 shows the hierarchy of roles.

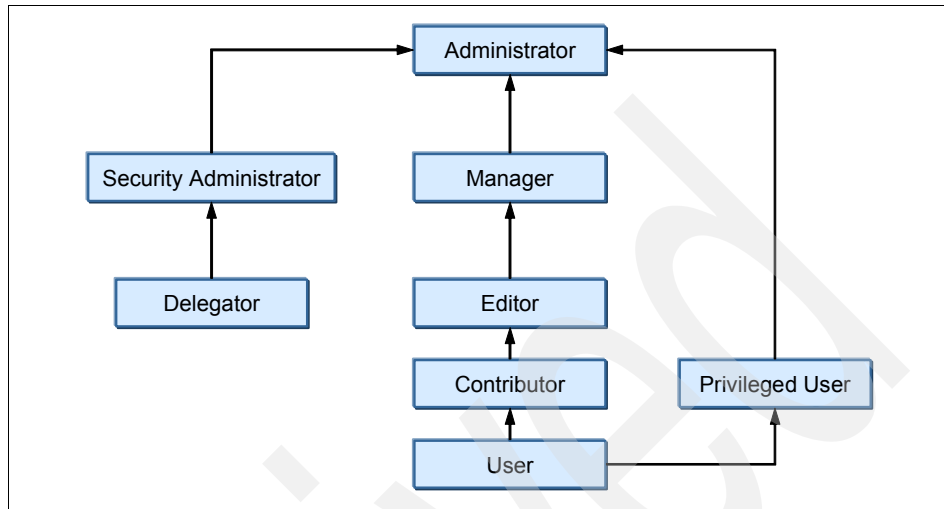


Figure 3-3 WebSphere Portal Server role hierarchy

Further information about role hierarchy and an explanation of each of the roles, see the “Roles” topic in the WebSphere Portal Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/security/sec_roles.html

In the LGI scenario, users are not permitted to modify any settings for the portlets. Therefore, the User role is granted to all users. Portal administrators are granted the Administrator role to the new portlets automatically because of the inheritance and propagation of roles. The access control configuration for the LGI scenario is configured by using the Resource Permissions Portlet, which is one of the portal administration portlets. The XML configuration interface and the portal scripting interface can also be used to complete the configuration.

The desired roles must be granted to users on the following objects:

- ▶ The Web module for the deployed portlets
- ▶ The portlet application definition, the individual portlets, or both

Granting the roles on the portlet application causes all of the portlets to be granted the same roles because of inheritance and propagation. In some circumstances, it might be necessary to grant different roles to each of the portlets. This can be achieved by setting roles only on the portlets or by settings roles on the portlet application definition in combination with role blocks and roles

granted to those portlets that require different settings. The role and resource hierarchy provides a powerful access control system that can reduce the administrative overhead of security configuration.

After the security configuration is completed, the last task is to create pages on which the portlets will be placed. In the LGI scenario, a top level LGI page was created with the following pages beneath it:

- ▶ Welcome
- ▶ New Quotes
- ▶ Your Quotes
- ▶ Your Policies
- ▶ Your Details

For detailed information about creating pages, see the “Creating pages” topic in the WebSphere Portal Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/admin/mp_create_pages_c.html

The portlets are placed on the page with the same name by using the page layout tools. Each page consists of a single column container with the appropriate portlet placed in that container. For detailed information about page layout and adding portlets to pages, see the “Layout and content” topic in the WebSphere Portal Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wpdoc/v6r1m0/topic/com.ibm.wp.ent.doc/admin/admcustom_layout.html

Finally, appropriate access control settings are applied to the pages to allow users to access the pages and view the portlets. The only role granted on the pages is the User role, which allows users to see the pages and portlets but not to modify the layout of the page. As with the access control settings for the portlets, the access control settings for the pages are configured by using the Resource Permissions portlet.

3.4 WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z

Several steps are necessary to configure the WebSphere Application Server 6.1 Network Deployment to meet the needs of the LGI scenario. The high level steps are explained in this section. For more information, see “Instructions for 3.4, ‘WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z’” on page 169.

3.4.1 Creating the configuration

The LGI Scenario environment uses WebSphere Application Server 6.1 Network Deployment to host a Web application. This environment consists of a cell that contains a deployment manager and three nodes. Each node has an application server that is a member of the cluster. This configuration is used to provide high availability and load balancing.

The creation of this configuration on AIX entails the following high-level steps:

1. Install WebSphere Application Server Network Deployment on servers A through D.
2. Create a deployment manager profile on server A.
3. Create custom profiles on servers B through D.
4. Federate nodes on servers B through D to the deployment manager on server A.
5. Use the deployment manager to create a cluster of servers.

For the detailed steps that are required to install and configure WebSphere Application Server 6.1 Network Deployment, see the following Information Center topics:

- ▶ “Installing your application serving environment”
<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/welc6topinstalling.html>
- ▶ “Setting up the application serving environment”
<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/welc6topserver.html>

3.4.2 Configuring security

The following steps describe the high-level process to secure the WebSphere Application Server 6.1 cell:

1. Configure a user repository.
2. Enable administrative security.
3. Enable application security.
4. Enable Java 2 security.
5. Enable manual node synchronization.
6. Use SSL between the Web server and WebSphere Application Server.
7. Use SSL between LDAP and WebSphere Application Server.
8. Change the WebSphere Application Server process identity.

For more information, see “Instructions for 3.4, ‘WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z’” on page 169.

It is important that all node agents in your cell are started before you make any changes to security settings. Any node agents that have not been started before changes are made to the security settings might require a manual synchronization with the deployment manager before they will operate correctly.

User account repository

WebSphere Application Server can use the following types of user account repository:

- ▶ Federated repositories
- ▶ Local operating system
- ▶ Standalone LDAP registry
- ▶ Standalone custom registry

The LGI scenario uses a standalone LDAP registry as its user account repository for WebSphere Application Server running on AIX. Further information about the LDAP structure associated with this example, see 3.2.2, “Creating the LDAP structure” on page 54.

Administrative security

Administrative security is used to restrict access to the administrative interfaces of WebSphere Application Server. Administrative security is also a prerequisite to application security. When administrative security is enabled, it is possible to grant different levels of administrative access to users and groups. The following types of administrative roles, among others, are supported:

- ▶ The *Administrator* role has full administrative access to the application server.
- ▶ The *Operator* role can change the run time state of resources such as applications and servers. In addition, it can perform any tasks allowed by the Monitor role.
- ▶ The *Configurator* role can modify the configuration of the application server and can perform any tasks allowed by the Monitor role.
- ▶ The *Monitor* role can view the configuration and current state of the application server.
- ▶ The *iscadmins* role can manage users and groups within the administrative console.

These roles can be granted to any user or group defined within the currently active user account repository. Where possible, assigning roles to groups rather than individual users can simplify administration.

Application security

Application security is used to secure applications and to enable authentication and authorization of application users. Application security is disabled by default and can only be enabled when administrative security is enabled. The LGI applications have various security roles defined and require that application security is enabled for correct operation.

Java 2 security

Java 2 security is used to restrict the level of access that applications have to resources. Applications are granted access to resources by using policy files. The LGI scenario does not use Java 2 security. Therefore, Java 2 security was not enabled.

Enabling Java 2 security on an environment that contains applications without policy files can cause problems when running the applications. This occurs when applications require access to restricted resources but have not been granted the appropriate permissions. The applications will fail to run correctly until the permissions have been granted.

Manual node synchronization

Manual node synchronization allows a node agent to retrieve a copy of the configuration from the deployment manager. This is useful in situations where there is a problem with the configuration and the node agent either cannot start or cannot communicate with the deployment manager. An example of such a situation is when the security settings for the cell have been modified while a node agent was stopped.

WebSphere Application Server process identity

In situations where WebSphere Application Server was installed by a root user, the process runs as root and effectively gives the WebSphere administrator root access to all machines in the cell. To provide better security, run WebSphere processes as a non-root user that is restricted to only the operating system resources that are required to run WebSphere Application Server. In an environment with multiple servers, set all WebSphere Application Server processes to run as the same user.

After the ownership of files is changed, it is possible to start the process by using the non-root user. It is still possible to start the process as a root user, and the process will run under the root ID. This can cause problems when a non-root user attempts to start processes. Any new log files will be owned by the root user, and the non-root user does not have permission to write to these files, which causes the server to fail to start. This problem can be avoided by configuring WebSphere Application Server to always run as the desired non-root user even when started by root.

Whenever you change the process execution settings for a particular component, restart that component to ensure that the new settings are active.

SSL between LDAP and WebSphere Application Server

When WebSphere Application Server is configured to use LDAP as its user account repository, it sends user names and passwords to the LDAP server for authentication. Communication with an LDAP server is not secure by default. Therefore, a network sniffer can be used to obtain user names and passwords that are sent between WebSphere Application Server and the LDAP server. SSL is used to protect the link between the two servers.

Communication over SSL uses SSL configurations (also known as *repertoires*) that define which key and truststores to use for communication. It is possible to use the default SSL configurations, keystores, and truststores that were created when WebSphere Application Server was installed. In the LGI scenario, new SSL configurations, keystores, and truststores were created to verify the specific actions required for configuration.

SSL between the Web server and the application server

SSL can be configured between the Web server and application server to prevent attempts to intercept the data that is transmitted. Additionally, client authentication between the Web server and application server can be used to limit WebSphere Application Server access to the Web server. This ensures that all requests to WebSphere Application Server must be routed through the environment as expected and helps to prevent attempts to bypass layers of security.

3.4.3 Deploying the application

To deploy the LGI front-end applications, you must define the required DB2 and WebSphere MQ resources in the WebSphere Application Server administrative console. The following section describes those tasks at a high level. For detailed information about these tasks, see the documentation in the WebSphere Application Server Network Deployment Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.webSphere.nd.doc/info/welcome_nd.html

To deploy the LGI front-end applications to the application server, follow these steps as described in detail in the information center documentation:

1. Define a DB2 JDBC provider.
2. Define a JDBC type 4 data source under this provider for Quote Accept.

3. Create the Java Message Service (JMS) resources that are required by the applications, which include the following resources:
 - Queue connection factories
 - Queues
4. Install the application enterprise archive (EAR) files.

3.5 WebSphere MQ on AIX and Linux for System z

This section describes the configuration of WebSphere MQ 6.0 on AIX or Linux for System z to meet the needs of the LGI scenario. WebSphere MQ 6.0 is similar on the two different platforms or any UNIX-based platform. The product is installed, configured, and administered in much the same way. Therefore, the information in the following sections applies to both platforms unless explicitly stated otherwise.

3.5.1 Configuring

The LGI scenario environment consists of a WebSphere Application Server and Portal cluster that hosts the Web application. Each machine that hosts a server within this WebSphere Application Server or Portal cluster has a local WebSphere MQ queue manager. These queue managers are added to a WebSphere MQ cluster with the WebSphere Message Broker queue managers on z/OS (which contain the full repositories).

On each of the machines hosting a server within either the WebSphere Application Server or Portal Cluster:

1. Install WebSphere MQ as explained in the “Installing a WebSphere MQ server” topic in the WebSphere MQ Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.amqaac.doc/aq10190_.htm

- On AIX, while following the instructions, select the following SSL file sets:
 - mqm.keyman.rte
 - gskta.rte
 - gksa.rte
- On Linux for System z, while following the instructions, select the following SSL packages:
 - gsk7bas64-7.0-4.14
 - MQSeriesKeyMan-U814340-6.0.2-4

2. Apply the latest available fix pack as explained in the instructions in the “Applying service” topic at the following address:
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.amqaac.doc/aq10720_.htm
3. Create a queue manager by using the instructions as explained in the “Creating a queue manager” topic at the following address:
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.amqzag.doc/fa10850_.htm
4. Add the queue manager to the existing WebSphere MQ cluster. For information about WebSphere MQ Clustering, see the “Getting started with queue manager clusters” topic at the following address:
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzah.doc/qc10220_.htm
5. Define the queues that are required by the application to the queue manager.

3.5.2 Security configuration

The LGI Web application is deployed to WebSphere Application Server and connects to a WebSphere MQ queue manager on the same machine (or logical partition (LPAR)) by using the bindings mode (defined on the WebSphere MQ queue connection factory definition). The application uses JMS to send messages to queues on remote cluster queue managers and receive messages from queues on local queue managers.

Figure 3-4 illustrates this architecture.

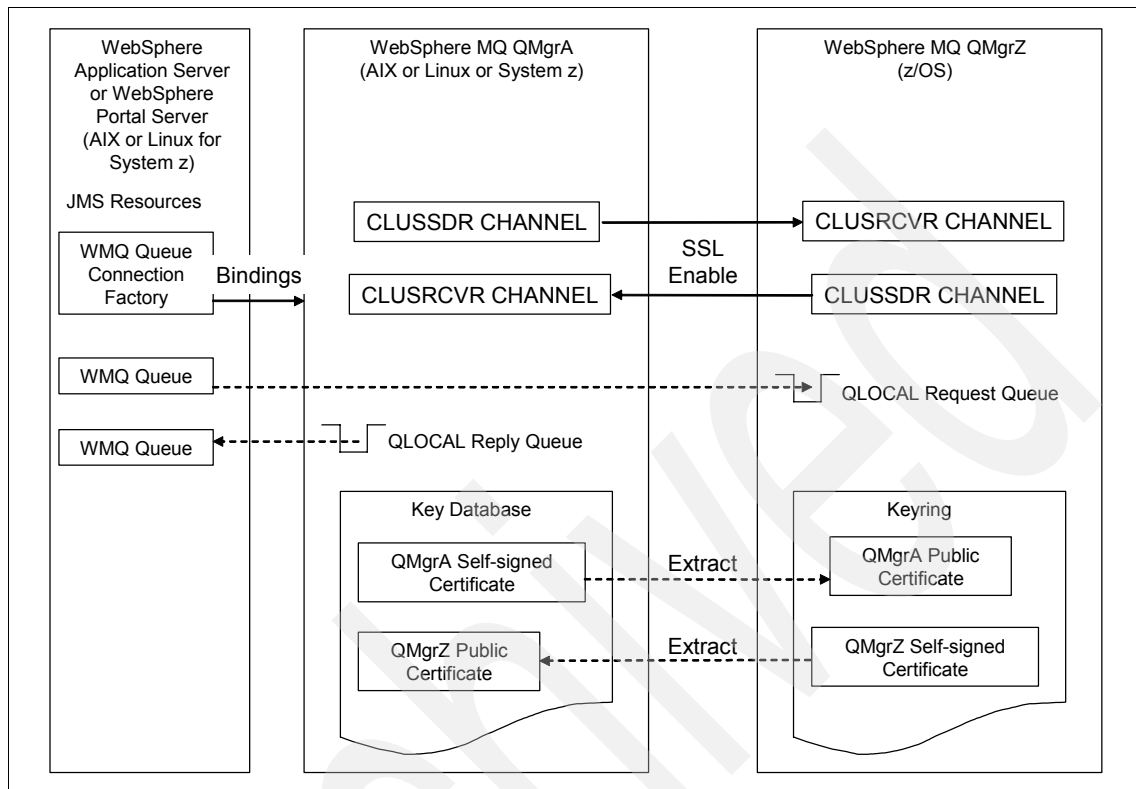


Figure 3-4 Logical LGI WebSphere MQ security architecture

The following WebSphere MQ security was configured for the LGI scenario environment AIX and Linux for System z queue managers:

- ▶ Access control
- ▶ SSL

Configuring access control

When WebSphere Application Server connects to WebSphere MQ by using the bindings mode, WebSphere MQ authentication and authorization are performed against the user that is currently logged on for the WebSphere Application Server process. On UNIX platforms, WebSphere MQ access control is granted at the group level rather than at the user level.

To configure access control:

1. Log on to an operating system session as the WebSphere MQ administrator.
2. Grant connect and inquire access to the queue manager for the group.
3. Grant put access to the cluster transmission queue for the group, allowing the WebSphere application to put messages to request queues on remote cluster queue managers.
4. Grant get, inquire, and browse access to all local reply queues for the group, allowing the JMS receive to obtain the reply messages.
5. Review access granted to the queue manager resources.
6. Review access to the queue manager resources granted to the group.
7. Refresh the queue manager's security configuration for the changes to take effect.
8. Log off the WebSphere MQ Administrator.

In addition to securing local access control, secure each queue manager for remote access. Any channel without an MCAUSER value allows user impersonation and anonymous administrative authority. The LGI front-end and back-end queue managers do not share a single security domain. Therefore, using context channel security (by setting the PUTAUT channel attribute to CTX) can be difficult to maintain. As a result, default channel security is used (by setting the PUTAUT channel attribute to DEF) in conjunction with MCAUSER (set to a user with low-privileged group).

For step-by-step instructions, see “Instructions for 3.5, ‘WebSphere MQ on AIX and Linux for System z’” on page 183.

Debugging access control problems

WebSphere MQ trace can provide useful information (the user or group, resource, and authority level required) to diagnose access control problems.

To run a WebSphere MQ trace on AIX:

1. Log on to an AIX session as the WebSphere MQ administrator.
2. Start the WebSphere MQ trace.
3. Recreate the access control problem.
4. Stop the trace.
5. Format the trace output.
6. Open the formatted trace file and search for the phrase “insufficient authority” (associated with the MQOPEN).
7. Log off the WebSphere MQ Administrator.

Secure Sockets Layer on MQ channels

The cluster receiver channel determines whether SSL is enabled on a channel and at what level through the use of the following properties:

- ▶ SSL CipherSpec (SSLCIPH)
- ▶ Authentication of parties initiating connections (SSLCAUTH)
- ▶ Accept only certificates with distinguished names that match these values (SSLPEER)

Note that this property is not covered in this project.

- ▶ Match certificates to local (certificate name filtering)

Note that this property is not covered in this project.

Keep in mind the following considerations:

- ▶ Specifying a SSLCIPH value enables SSL on the channel.
- ▶ Both ends of a channel (that is, the receiver and corresponding senders) must specify the same cipher specification value. Auto-defined cluster sender channels are based on the definition of remote cluster receiver channels.
- ▶ Not all ciphers are supported on all platforms. For more information about supported cipherspec values, see the “Specifying CipherSpecs” topic at the following address:

http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqzas.doc/sy12870_.htm

- ▶ When SSL is enabled on a channel, the receiver channel queue manager (server) must have its own private certificate in its own key database. In addition, all corresponding sender channel queue managers (clients) must have the exported public version of this certificate in their key databases.
- ▶ If client authentication has been enabled by specifying SSLCAUTH(REQUIRED), then all corresponding sender channel queue managers (clients) also require their own private certificates in their key databases. In addition, the receiver channel queue manager (server) requires the corresponding exported public certificates in its key database.
- ▶ Although self-signed certificates were used, using a certificate authority to sign certificates is the recommended approach for use in production environments.
- ▶ Although self-signed certificates were used, using a certificate authority to sign certificates is the recommended approach for use in WebSphere MQ cluster environments.

- The labels of certificates representing WebSphere MQ queue managers must use the following formats:
 - On distributed platforms, use `ibmwebsphermqqmgrname` all in lowercase.
 - On z/OS, use `ibmWebSphereMQQMGrName`, where `QMGrName` is uppercase.

In addition, when adding a public certificate to a key database or key ring, the label specified must match the label value that was specified when the certificate was created.

Configuring Secure Sockets Layer

In the LGI scenario environment, SSL is configured on the cluster sender and receiver channels between the queue managers that are used by the WebSphere Application Server or Portal servers that host the Web application and the z/OS queue managers used by the message brokers. In addition, client authentication is also enabled to prevent queue managers without a certificate from connecting to the cluster.

For instructions on configuring the end of the SSL channels that include the z/OS queue manager, see 4.5, “WebSphere MQ on z/OS” on page 113.

To enable SSL on sender and receiver cluster channels to and from a z/OS queue manager on a UNIX queue manager:

1. Obtain the public certificate for the remote (z/OS) queue manager.
2. Obtain the value of the queue manager's property SSL key repository (SSLKEYR). On distributed platforms this is set to `/var/mqm/qmgrs/QMGrName/ssl/key` by default, although the key repository does not exist until explicitly created.
3. Create a new key database by using a key management tool, for example `gsk7ikm` or `ikkeyman`.
4. Create a self-signed certificate for the local queue manager in the key database.
5. Extract the local queue manager's public certificate so that it can be added later to the remote queue manager's key ring.
6. Add the remote (z/OS) queue manager's public certificate to the key database.
7. Set the cipher specification on the sender channel to match the value specified on the equivalent receiver channel on the remote (z/OS) queue manager.
8. Set cipher specification to be used on receiver channel and request client authentication.
9. Refresh the queue manager's security settings to implement the changes.

10. Restart both ends of each cluster sender and receiver channel for SSL settings to be implemented, including the dynamic cluster sender channels. These are defined automatically with the same SSLCIPH value as the corresponding receiver channel after they are restarted.

Although these steps explain how to set up SSL certificates for communication between two queue managers, in practice, when using self-signed certificates, each queue manager must share its public certificate with every other queue manager in the cluster. This results in complex SSL certificate administration processes. Therefore, use certificate authorities in WebSphere MQ cluster environments.

For step-by-step instructions, see “Instructions for 3.5, ‘WebSphere MQ on AIX and Linux for System z’” on page 183.

3.5.3 Application deployment

No application deployment is required for WebSphere MQ on AIX or Linux for System z.

3.6 DB2 Enterprise 9 on AIX and Linux for System z

We used IBM DB2 Enterprise Server Edition, Version 9.1, for our AIX and Linux for System z environment product and user databases. We created each database on its own file system and conducted weekly housekeeping and backups. DB2 is similar on the two different platforms or any UNIX-based platform. The product is installed, configured, and administered in much the same way. Therefore, the information in this section applies to both platforms unless explicitly stated otherwise.

3.6.1 Installing DB2 Enterprise 9

To install DB2 Enterprise 9:

1. Review the release notes, which can be found in the DB2 Version 9 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.doc/doc/c0023859.htm>

2. Download the product code from either Passport Advantage® or XL Software Downloads:
 - For AIX, obtain electronic software download part number C92AZML – DB2 Enterprise Server Edition V9.1 - CPU Option (64-bit) and Fix Pack 4 for DB2 v9fp4_aix_ese.tar.gz.
 - For Linux on System z, obtain DB2 Enterprise Server Edition V9.1 for Linux on System z9® and zSeries® Multilingual (C13KVML) and Fix Pack DB2_ESE_V913_LNXS390X.tar.
3. Install DB2 by using the setup wizard. Install DB2 on a machine designated as the database server for products that support a remote database. Not all products support remote databases. For these products, you must install DB2 on the same machine as the product. For instructions to install DB2 servers, see the DB2 Version 9 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.uprun.doc/doc/t0008875.htm>

The information center instructions also explain how to create a new DB2 instance with an owner. This user ID and password must be supplied by any software or administrator attempting to access the databases.
4. Install the DB2 license:
 - a. Extract the db2ese_c.lic license file from the downloaded code and copy it to the home directory of the instance owner that you just created.
 - b. Substitute the user ID that corresponds to the owner of the instance:

```
su - db2inst2
```
 - c. Install the license:

```
db2licm -a db2ese_c.licm
```
 - d. List the license that is installed:

```
db2licm -l
```

This command displays the type of CPU and the date of Permanent.

3.6.2 Configuration

The DB2 environment might need specific configuration for products that require a database. Each product installation or configuration specifies these settings. If more than one product requires a change to a specific setting, then the highest value specified is used to satisfy the product that has the greatest requirement.

Tip: For Linux on System z, it is necessary to increase the memory to 2 GB in order for DB2 to perform optimally.

3.6.3 Security configuration

The user ID and password of the instance owner must be supplied by any product or administrator that accesses the environment or a database.

3.6.4 Defining databases

To define your databases:

1. Create all databases under the home directory of the instance owner unless otherwise specified.
2. Create a separate file system for each database so that you can administer space more effectively, do your housekeeping easily, and have better control for each database.
3. Create the database. When creating the database, specify that the database be located on the file system created in step 2 by using the following command or command within a script:

```
CREATE DATABASE database1 on /database1
```

For the detailed instructions to create a file system, see “Instructions for 3.6, ‘DB2 Enterprise 9 on AIX and Linux for System z’” on page 192.

3.7 IBM HTTP Server 6.1 on AIX

The LGI scenario uses IBM HTTP Server as the Web server. The Web server is placed in the demilitarized zone (DMZ). The DMZ is separated from the internal network by firewalls. All requests to the other systems in the environment must be routed through the Web server, which helps to prevent attackers from accessing the more sensitive servers. The primary function of the Web server is to route incoming requests to the appropriate application server. The WebSphere plug-in for IBM HTTP Server is installed on the Web server and is used to route requests to the application servers.

3.7.1 Configuration

IBM HTTP Server 6.1 can be installed by following the instructions in the WebSphere Application Server Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.WebSphere.ihs.doc/info/ihs/ihs/tihs_installihs.html

The Web server plug-in for WebSphere Application Server can be installed by following the instructions in the WebSphere Application Server Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.WebSphere.nd.doc/info/ae/ae/tins_Webplugins.html

3.7.2 Security configuration

To secure IBM HTTP Server and the WebSphere Web server plug-in, you must enable HTTPS communication with browsers and SSL between the plug-in and WebSphere Application Server.

Enabling HTTPS communication with browsers

In the LGI scenario, personal information is transmitted between the users' browsers and the LGI environment. This personal information includes names, dates of birth, addresses, user names and passwords. Encryption ensures that this information cannot be changed or intercepted while in transit.

In the following example, self-signed certificates are used. In a production system, obtain appropriate certificates from a certificate authority to prevent users from seeing warnings about certificate problems.

To create a CMS key containing a self-signed certificate:

1. Open the key management GUI.
2. Create the key database file.
3. Create a self-signed certificate for IBM HTTP Server in the key database.

To enable HTTPS:

1. Add an entry to the `httpd.conf` file to configure the Web server to listen on port 443.
2. Add entries to the `httpd.conf` file to configure the Web server to use SSL on port 443 using the CMS database created previously.
3. Restart the HTTP server.

For more information, see "Instructions for 3.7, 'IBM HTTP Server 6.1 on AIX'" on page 194.

After you complete these steps, create a new virtual host in WebSphere Application Server. This virtual host is used to tell the application server to accept HTTPS requests from any host on port 443, which is necessary for requests to be properly routed to the application.

Enabling SSL between the plug-in and application server

In the LGI scenario, the Web server is located in the DMZ. A firewall blocks all traffic other than HTTP using port 80 and HTTPS by using port 443 from entering the DMZ. The traffic between the Web server plug-in and the application server can be further protected by enabling SSL communication. It is possible to restrict communication with the application server to known Web servers. This is achieved by enabling client authentication in the Web container for the application server.

To secure the link between the Web server plug-in and the application server:

1. Create a CMS database and self-signed certificate for the Web server plug-in.
2. Extract the public key from the CMS database.
3. Import the public key into the Web container truststore in WebSphere Application Server.
4. Extract the Web container public key from the keystore in WebSphere Application Server.
5. Import the Web container public key into the CMS database.
6. Configure the plug-in to use the CMS database.

3.8 WebSphere Application Server 6.1 Network Deployment on z/OS

This section describes the configuration of WebSphere Application Server 6.1 on z/OS to support the LGI application. The pure z/OS implementation of this application uses WebSphere Application Server components that are dedicated to support logical front-end and back-end components, as follows:

- ▶ The LGI applications
- ▶ WebSphere Process Server
- ▶ WebSphere Service Registry and Repository

The WebSphere Application Server environment is configured in a Network Deployment topology to provide high availability and load balancing. This allows the z/OS Workload Manager to manage application server workload across multiple LPARs in a z/OS Parallel Sysplex environment.

The configuration described here uses WebSphere Application Server clusters. *Clusters* are sets of servers that are managed together and participate in workload management. Clusters enable enterprise applications to scale beyond the amount of throughput capable of being achieved with a single application server. Clusters also enable enterprise applications to be highly available because requests are

automatically routed to the running servers in the event of a failure. The servers used by the LGI application are clustered across different host machines.

3.8.1 Configuration

WebSphere Process Server and WSRR are installed into predefined WebSphere Application Server environments. We dedicated a Network Deployment environment to these functions, and therefore, had three Network Deployment cells:

- ▶ A1 for WebSphere Process Server
- ▶ A2 for WebSphere Application Server applications
- ▶ A3 for WebSphere Service Registry and Repository

Our configuration strategy involved dedicating a cluster to an application so that applications are deployed at the cluster level. If an LPAR becomes unavailable, a cluster server on another LPAR can perform the same work with minimal updates to WebSphere MQ and JDBC resource definitions. These resources can be configured by using a dynamic virtual IP address (VIPA) and shared queues to eliminate changes required when running the application on different LPARs.

Figure 3-5 shows the layout of the A2 cell and how the LGI applications are installed into clusters.

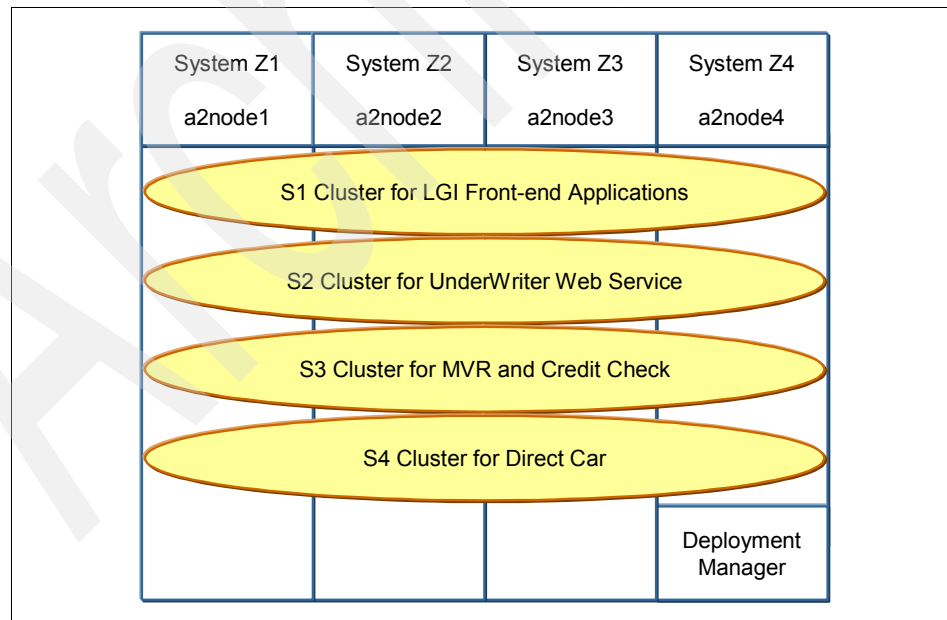


Figure 3-5 WebSphere Application Server network deployment topology

Although the applications are available by using any server in the cluster, we ran servers on particular systems at a given time rather than to have all servers running. We did this to ensure that work was distributed among the LPARs in a manner that fit well with the other work that was being done on those systems. For more information about the WebSphere Process Server and WSRR configurations, see 4.3, “WebSphere Process Server on z/OS” on page 103, and 4.4, “WebSphere Service Registry and Repository on z/OS” on page 110.

3.8.2 Tools to assist in configuration

zPMT is a tool that can be used when planning the Network Deployment cell for z/OS. A sample spreadsheet that can be used to generate response data for zPMT is available for download at the following address:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1331>

The spreadsheet is divided into separate “worksheets” within the file: one for the Deployment Manager, one for a Standalone Server, and so on. Read the instructions on the first worksheet tab to understand how the spreadsheet works. Then complete the variables on the second worksheet tab. The values that you enter are applied through the document to the other worksheets to create the variables. For additional documentation about zPMT, see “Introducing the zPMT Configuration Tool for WebSphere z/OS” the following address:

<http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100871>

After you complete the spreadsheet with the desired values, the data is extracted into a response file that is read by the WebSphere Application Server Toolkit V6.1.1. You might need to update the generated response values from within the WebSphere Application Server Toolkit if the resulting names do not meet local naming conventions. After the responses are read into the WebSphere Application Server Toolkit and results are verified, configuration job control language (JCL) is generated and uploaded to z/OS. The jobs and instructions look similar to those generated by using the WebSphere Application Server ISPF `bbowstrt clist` in the `sbboclib` data set.

3.8.3 Security configuration

The following types of security and functions are necessary secure the z/OS WebSphere Application Server 6.1 cell by using Resource Access Control Facility (RACF) profiles:

- ▶ Administrative security
- ▶ Application Security
- ▶ Manual node synchronization

Start all node agents in the cell before changing security settings. Any node agents that have not been started when changes are made to the security settings might require a manual synchronization with the deployment manager before they will operate correctly.

Administrative security

The JCL generated by WebSphere Application Server Toolkit includes REXX executables that can be used to define the RACF profiles that in turn are required to secure the administrative console and define roles for users. The jobs that are generated to define the RACF profiles are run prior to enabling security in the administrative console.

Administrative security is enabled by selecting the Enable administration security check box under “Secure administration, applications, and infrastructure” (Figure 3-6).



Figure 3-6 Enabling administrative security in the Administrative Console

The roles that are defined here are ignored if the System Authorization Facility (SAF) authorization is enabled. If SAF is chosen, the ejbrole RACF class is used to control user and group access. If you plan to qualify the roles by a domain name, you must have specified this domain name during the generation of the configuration JCL in WebSphere Application Server Toolkit.

Our setup uses RACF to control authorization to resources in the administrative console. This is supported by the following RACF ejbrole profiles:

- ▶ administrator
- ▶ monitor
- ▶ configurator
- ▶ operator
- ▶ deployer
- ▶ adminsecuritymanager

Enabling application security

Application security is used to secure applications and to enable authentication and authorization of application users. This function is disabled by default. Application security can only be enabled when administrative security is enabled.

To enable application security:

1. Log in to the Integrated Solutions Console on the deployment manager.
2. Select **Security**.
3. Select **Secure administration, applications, and infrastructure**.
4. Verify that **Administrative security** is enabled.
5. Select **Application security**.
6. Click **Apply**.
7. Click **Save**.
8. Ensure that the **Synchronize changes with Nodes** option is selected (Figure 3-7).

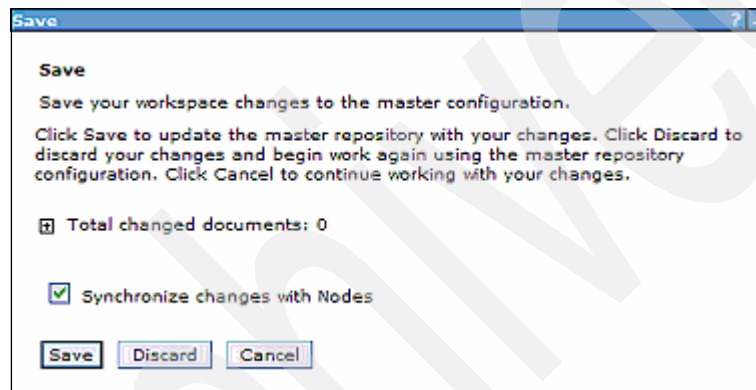


Figure 3-7 Saving changes to the configuration

The LGI applications have various security roles defined, and the applications require that application security is enabled for correct operation. These roles are defined to the ejbrole RACF class and include the following roles:

- ▶ AllQuoteUsers
- ▶ StaffDiscount
- ▶ EJBQuoteUser

The roles used by an application can be viewed in the administrative console by looking at the application security role to user/group mapping as shown in Figure 3-8 on page 85. These roles are case sensitive. Therefore, the RACF ejbrole profile must match the case shown in Figure 3-8 on page 85 for the application.

RACF does not accept blank spaces between profile names. For example, “All Quote Users” is not be allowed as an RACF profile, but “AllQuoteUsers” is acceptable.

Figure 3-8 shows the security roles that are required by the application. These roles are defined during application development in the WebSphere Integration Developer tool. When the application having security roles defined to it is deployed as a WebSphere application, its roles can be viewed in the administrative console. The roles can be found by navigating to **Applications** → **Enterprise Applications** → **LGI SecureQuote** → **Security role to user/group mapping**. When using the RACF class EJBROLE to control authentication, the users do not need to be mapped in the administrative console.

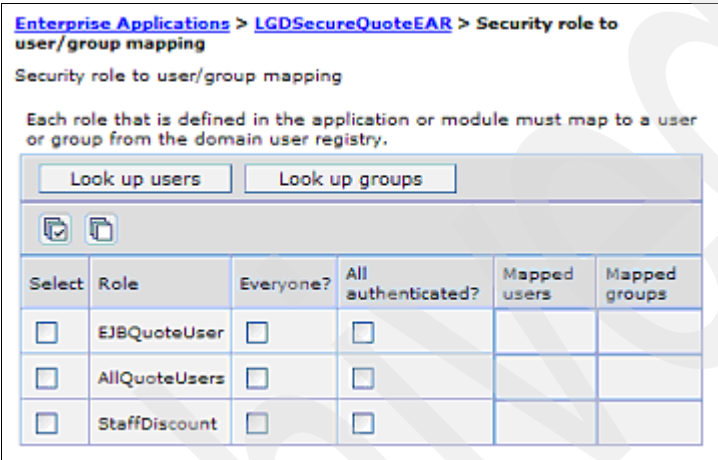


Figure 3-8 LGI Secure Quote application security role mappings

3.8.4 Application deployment

To deploy the LGI front-end applications, you must define the required DB2 and WebSphere MQ resources in the administrative console. In this section, we describe those tasks at a high level. For detailed information about these tasks, see the WebSphere Application Server Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.WebSphere.zseries.doc/info/welcome_nd.html

To deploy the LGI front-end applications into WebSphere Application Server, follow these steps as described in detail in the information center documentation:

1. Define a DB2 Universal Database non-XA JDBC provider.
2. Define a JDBC type 4 data source under this provider for Quote Accept.

3. Create the JMS resources required by the applications, which include the following resources:
 - Queue Connection Factories
 - Queues
4. Install the application EAR files.

3.8.5 JDBC resource definitions

This section describes the type of JDBC data source connections defined in WebSphere Application Server to access the DB2 tables used by the LGI application.

Connection types

There are two types of JDBC connections: type 2 and type 4. Type 2 JDBC connections are used for a local connection to DB2 by using Resource Recovery Services (RRS). This type connects an application, such as WebSphere Application Server, to a local DB2 running on the same LPAR.

Type 4 JDBC connections are used to connect to remote DB2 subsystems by using TCP/IP and the DB2 Distributed Data Facility (DDF). This type of connection is used when the application connecting to DB2 resides on a different system than where DB2 is running or if the TCP/IP Sysplex Distributor is used to route requests to any available DB2 in a data-sharing environment. In the latter setup, the z/OS Workload Manager routes work to the system with the greatest capacity to handle the work.

These JDBC connections are configured in WebSphere Application Server for a DB2 Universal JDBC Driver Provider. Our environment used type 2 connections to access the Direct Car database and type 4 connections to access the LGI quote/accept database. The Direct Car database and application are set up on the same LPAR. Therefore, a type 2 connection is appropriate.

Using type 2 connections over type 4 connections has the following performance advantages:

- ▶ CPU cost

When the type 2 JDBC driver is used, the CPU cost of marshalling and de-marshalling data into a wire format and the associated TCP/IP flows is eliminated.

- ▶ No task re-dispatch

When using the type 2 JDBC driver, there is no task re-dispatch and the DB2 activity continues on the current thread of execution. The priority or importance of the current request being processed is maintained.

- RRS for superior performance

Type 2 JDBC driver uses RRS for managing the transaction state, which is a superior performer than the type 4 (XA) JDBC provider.

A type 4 connection is used for the quoteAccept data source to take advantage of the Sysplex Distributor capabilities to access a DB2 data-sharing member on any LPAR. A distributed VIPA is defined to distribute requests to any member of the DB2 data sharing group. For additional information about the Sysplex Distributor, see the *z/OS V1R10.0 Comm Svr: IP Configuration Guide*, SC31-8775-13.

Security for DB2 access

An RACF user ID is defined for access to the DB2 data sharing group. This user ID is specified as the component-managed authentication alias in the data source definition. RACF profiles protect access to DB2 and authority is granted to the user ID specified for the JCA alias appropriate to the functions it needs to perform in DB2.

Security for WebSphere MQ access

RACF profiles also protect the WebSphere MQ resources by using the WebSphere Application Server server-started task user ID to authenticate. See 4.5.2, “Security configuration” on page 115, for additional information.

Generating the DB2 back-end ID

The EJB within the LGI Quote front-end Web application use EJB 2.0 entity beans to access a DB2 database. During application development, code is generated to perform the runtime actions for DB2 Universal Database V8.2 for Linux, UNIX, and Windows. This code is referred to as the *back-end ID*. To run this application successfully when connecting to a DB2 z/OS database, appropriate back-end code must be generated during application deployment by using the following steps if installing the application by the Integrated Solutions Console:

1. In the “Preparing for the application installation” step, select **Show me all installation options and parameters**.
2. In the “Select installation options” step, select **Deploy enterprise beans**.
3. In the “Provide options to perform the EJB Deploy” steps, select the appropriate value from the Database type list and enter the schema or qualifier name of the tables for the Database schema.

If installing the application by using a wsadmin Jython script, in the attributes passed to the **AdminApp.install** command, include the following parameters:

```
deployejb -deployejb.dbtype dbTypeValue -deployejb.dbschema  
dbSchemaValue
```

For supported values and further details, see “Options for the AdminApp object install, installInteractive, edit, editInteractive, update, and updateInteractive commands” in the WebSphere Application Server Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/rxml_taskoptions.html

An alternative approach is to ensure that the appropriate back-end ID code is included inside the application EAR file before the application is deployed. This alternative approach can be achieved using the `ejbdeploy` command. For details, see the “The ejbdeploy command” topic in the WebSphere Application Server Version 6.1 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.etools.ejbbatchdeploy.doc/topics/regenc.html>

To check which back-end ID has been generated for an EAR file:

1. Obtain the enterprise application EAR file used in the runtime environment:
 - a. From the Integrated Solutions Console, select **Applications**.
 - b. Select **Enterprise Applications**, select the appropriate application, and click **Export**. The EAR file is displayed. Right-click the file name, and use the browser’s save function to save the file to a local temporary directory.
2. Use an extraction utility to open the EAR file and expand the JAR file of the EJB within it. (This might be easier on a Windows PC rather than AIX or z/OS).
3. Sort the contents by path name, locating files with paths of the format *packageName/websphere_deploy/backendID*. Note that the default *backendID* value for applications developed in Rational Application Developer 7.0 is `DB2UDBNT_V82_2`.

3.9 WebSphere MQ on z/OS

This section explains how to configure WebSphere MQ 6.0 to meet the needs of the LGI scenario.

3.9.1 Configuration

The LGI scenario environment consists of a WebSphere Application Server cluster that hosts the Web application. Each machine that hosts a server within this cluster has a local WebSphere MQ queue manager. These queue managers

are part of a queue-sharing group that, along with the queue managers, is local to the CICS regions and to the message brokers.

On each LPAR:

1. Install WebSphere MQ as explained in “Installing WebSphere MQ for z/OS” in the WebSphere MQ 6.0 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqsat.doc/csqs82du.htm>
2. Create and customize the queue managers, channel initiators, and queue-sharing group as explained in “Customizing your queue managers” in the WebSphere MQ 6.0 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqsav.doc/csqsav0410.htm>
3. Define the queue-sharing group’s administrative structure and an application structure (LGISOA for our example) to be used by all shared queues that are used in this application.
4. Create and customize the WebSphere MQ cluster with the queue managers that are local to the message broker and the WebSphere Application Server or Portal servers that are hosting the front-end application. Follow the tasks as outlined in the *Queue Manager Clusters* book in the WebSphere MQ 6.0 Information Center at the following address:
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzah.doc/qc10120_.htm
5. Define the queues to be used by this application.

3.9.2 Security configuration

The LGI Web application is deployed to WebSphere Application Server and connects to a WebSphere MQ queue manager on the same z/OS LPAR by using bindings mode (defined on the WebSphere MQ queue connection factory definition). The application uses JMS to send messages to shared request queues and receives messages on local cluster queues. All queue managers used for this application reside in the same Parallel Sysplex environment and use the same RACF database as shown in Figure 3-9 on page 90.

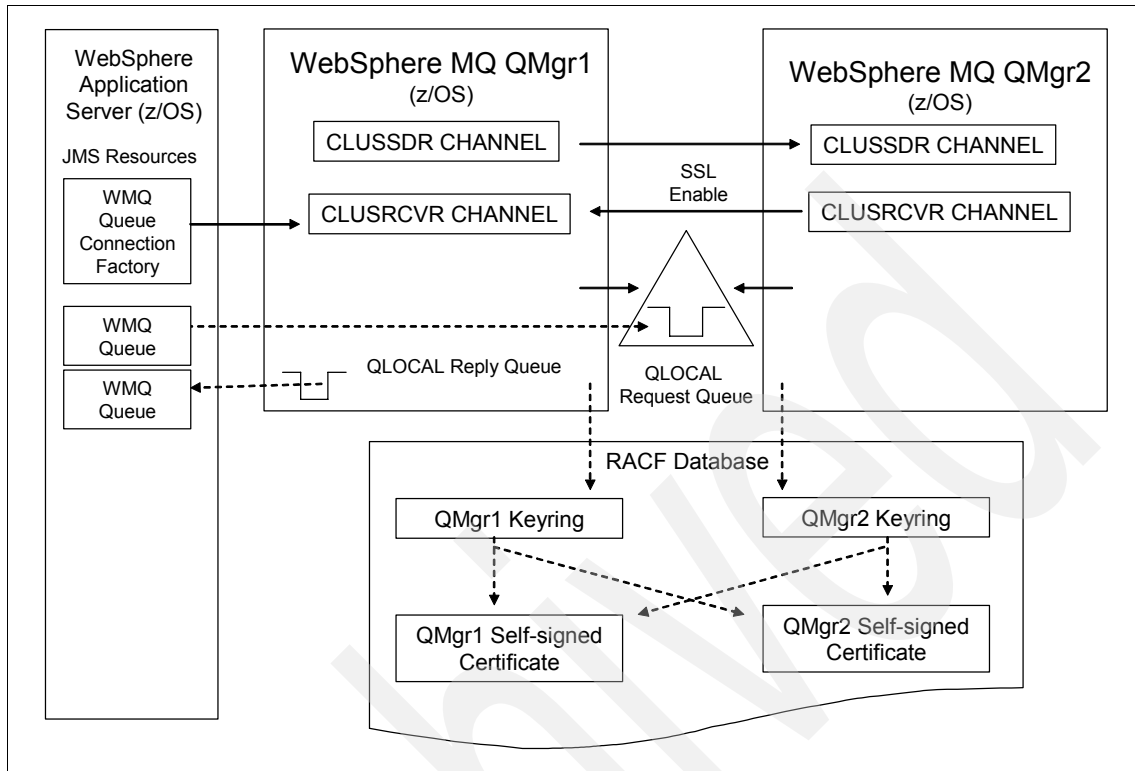


Figure 3-9 Logical LGI WebSphere MQ security architecture

The following WebSphere MQ z/OS security is configured for the LGI scenario environment:

- ▶ Access control
- ▶ SSL

Configuring access control

WebSphere MQ uses the SAF to route requests for authority checks to an external security manager (ESM). For our z/OS queue managers, we used RACF.

With RACF profiles, you can specify whether you want authority checks turned on or off for each queue manager individually (queue manager level security) or for every queue manager in the queue-sharing group (queue-sharing group level security). This is called *subsystem security*.

In our LGI scenario, WebSphere Application Server connects to WebSphere MQ by using the bindings mode. At this point, WebSphere MQ authentication and

authorization for the WebSphere Application Server started task are performed against the user who is currently logged on. On z/OS, we have RACF groups for the started tasks associated with the user IDs of the cell members. If all of the queue managers that you use for this scenario are part of the queue-sharing group, you can use queue-sharing group level security.

To provide sufficient access to the required WebSphere MQ z/OS queue manager resources for the LGI Web application:

1. Log on to a TSO session as with a user ID that is part of the RACF group that has administrative access for WebSphere MQ.
2. Define the RACF profile, and grant update access to all local reply and request queues for the group (allows the JMS receive to obtain the reply messages).
3. Refresh the queue manager's security configuration to implement the changes addressed.
4. Log off the WebSphere MQ Administrator.

For detailed instructions, see "Instructions for 3.9, 'WebSphere MQ on z/OS'" on page 195.

Configuring Secure Sockets Layer

In the LGI scenario environment, SSL is configured on the cluster sender and receiver channels between the queue managers used by the message brokers and the queue managers used by the WebSphere Application Server servers hosting the Web application.

The following steps are required to enable a z/OS queue manager to use SSL and to create a self-signed certificate for the queue manager. Note that this scenario does not use the Integrated Cryptographic Service Facility (ICSF) to store certificates. If ICSF is used, extra setup is required.

1. Grant the channel initiator address space user ID access to the RACF key rings.

The WebSphere MQ Information Center documentation directs users to grant read access. However, update access is required in the LGI scenario environment because the channel initiator user ID does not own the RACF key ring.

2. Enable the queue managers to use SSL.

The value must be greater than 1 but less than 50 to prevent storage allocation problems.

3. Restart the channel initiator address space (CHINIT) to implement the change.

4. Create the key ring for the queue manager.
5. Add the key ring to the queue manager by setting the queue manager's SSL key repository (SSLKEYR) property.
6. Create a self-signed certificate for the local queue manager.
7. Add the self-signed certificate to queue manager's key ring.
8. Refresh RACF to pick up the changes
9. Refresh the queue manager's SSL settings to implement the key ring changes.

For detailed instructions, see “Instructions for 3.9, ‘WebSphere MQ on z/OS’” on page 195.

Exchanging certificates with another z/OS queue manager

On a z/OS queue manager, to enable SSL on sender and receiver cluster channels to and from another z/OS queue manager:

1. Connect the remote (z/OS) queue manager's public certificate to the local queue manager's key ring.
2. Repeat the previous step with all other queue managers that will be part of this cluster until the local queue manager is connected to all the self-signed certificates of the other queue managers.
3. Refresh RACF to pick up the changes.
4. Refresh the queue manager's SSL settings to implement the key ring changes.
5. Set the cipher specification on the sender channel to match the value specified on the equivalent receiver channel on the remote (AIX) queue manager.
6. Set the cipher specification to be used on the receiver channel and request client authentication.
7. Restart both ends of each cluster sender and receiver channel in order for the SSL settings to be implemented.

Keep in mind the following considerations:

- ▶ RACF commands are issued from the ISPF TSO command interface while the user is logged on with sufficient RACF authority to create key rings and certificates.
- ▶ MQSC commands are issued from the ISPF WebSphere MQ utilities interface.

For detailed instructions, see “Instructions for 3.9, ‘WebSphere MQ on z/OS’” on page 195.

3.9.3 Application deployment

No application deployment is required for WebSphere MQ on z/OS.

3.10 DB2 Enterprise 9 on z/OS

The LGI scenario application uses DB2 Enterprise 9 for both the front-end and back-end components. In this section, we assume that DB2 is already installed on z/OS. We also assume that any configuration that needs to use WebSphere Application Server, WebSphere Process Server, WebSphere Service Registry and Repository, WebSphere MQ, and WebSphere Message Broker is complete.

This section contains details about the application databases only. Since the majority of the configuration is done to support the back-end functionality, you can find details about the overall environment and database design in Chapter 4, “Deployment of the back-end components” on page 99. The DB2 system programmer uses standard tools and utilities to define the databases that are needed. This section contains only information unique to the front-end server.

Figure 3-10 illustrates the basic configuration for the DBXG data sharing group. It is a four-way data sharing group with members DBX1, DBX2, DBX3, and DBX4. The LGIUSR9 database is also in this group.

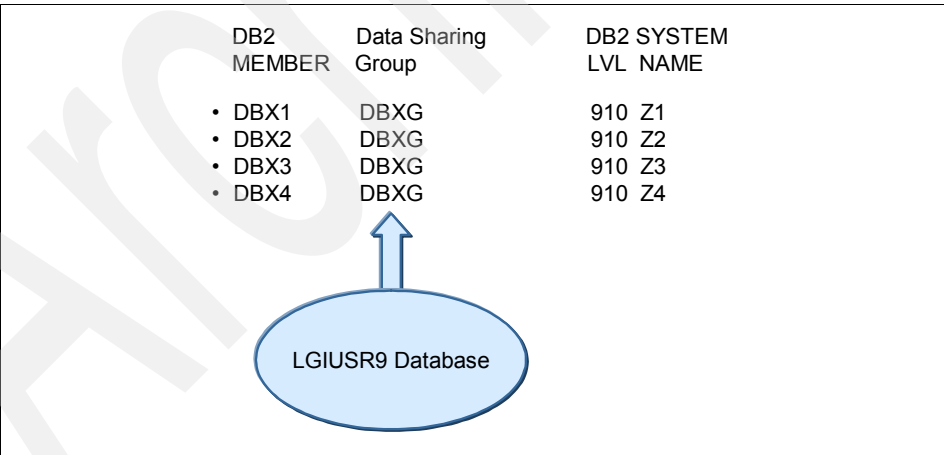


Figure 3-10 Data sharing group used for LGI

3.10.1 The LGI secure application configuration

Table 3-1 provides a high level description of the LGIUSR9 database that is used by the front-end application components.

Table 3-1 LGIUSR9 database description

| Database | Tablespace | Tables | Indexes | SG | BP |
|----------|------------|--------------------------|------------------|--------|-------------------|
| LGIUSR9 | LGIUSR01 | LGIUSERS.GENERATEQUOTEID | LGIUSERS.IQUOTE | LGIUSR | BP17(T) BP4(I) |
| | LGIUSR02 | LGIUSERS.USERDETAILS | LGIUSERS.IUSERID | | |
| | LGIUSR03 | LGIUSERS.STOREDQUOTES | LGIUSERS.ISTORE | | |
| | LGIUSR04 | LGIUSERS.POLICIES | LGIUSERS.IPOLICY | | |

3.11 WebSphere MQ File Transfer Edition on AIX

This section describes how to configure WebSphere MQ File Transfer Edition (FTE) 7.0 to meet the needs of the LGI scenario.

3.11.1 Configuration of an AIX agent

The LGI scenario environment contains a WebSphere Application Server and Portal cluster that hosts the Web application. Each of the front-end machines that hosts a server within either the WebSphere Application Server or Portal cluster has a local WebSphere MQ queue manager.

On each the front-end machines:

1. Install WebSphere MQ FTE. Choose the server install option so that binding connections to queue managers are supported.
2. Before creating a WebSphere MQ FTE agent, ensure that a coordination queue manager is available. See 4.9, “WebSphere MQ File Transfer Edition on AIX” on page 130, to install and configure the coordination queue manager.
3. After the coordination queue manager is configured at the back-end server, create a WebSphere MQ FTE agent.
4. Define the queues that are required by the agent on the local queue manager.
5. Check that the agent is registered with the coordination queue manager.

For step-by-step instructions, see “Instructions for 3.11, ‘WebSphere MQ File Transfer Edition on AIX’” on page 199.

3.11.2 Security configuration

WebSphere MQ FTE agents benefit from using the existing secured WebSphere MQ infrastructure. For example, the channels between queue managers already exist and are secured with SSL. Therefore, the WebSphere MQ FTE messages that are transmitted over those channels are secure.

WebSphere MQ FTE is a WebSphere MQ application. Therefore, appropriate steps are taken to enable access for WebSphere MQ FTE users. Each WebSphere MQ FTE agent runs associated with the user that executed the **fteStartAgent** command.

These users require access to WebSphere MQ objects including the following objects:

- ▶ The queue manager object
- ▶ Agent queues (named `SYSTEM.FTE.*.<agent name>`)
- ▶ The `SYSTEM.CLUSTER.TRANSMIT.QUEUE`

Security access is configured by using the RACF on z/OS and using the object authority manager (OAM) on AIX and Linux for System z.

In addition the basic security configuration described above, there are additional security configuration steps required in order to enable communication between agents and the coordination queue manager. For information about configuring the AIX coordination queue manager, see “Instructions for 3.11, ‘WebSphere MQ File Transfer Edition on AIX’” on page 199.

For more information about WebSphere MQ FTE security, see the following address:

http://www.ibm.com/developerworks/websphere/library/techarticles/0902_wyatt/0902_wyatt.html

3.11.3 Transferring files

The purpose of WebSphere MQ FTE is to transfer JPG files (uploaded by LGI customers) from the front-end machines to a back-end z/OS system. All files are transferred from a particular directory every hour by using the **fteCreateTransfer** command.

WebSphere MQ FTE provides graphical tooling for administration and views of the file transfer status for auditing purposes. WebSphere MQ FTE status information is published by the coordination queue manager, which allows products and applications to subscribe and receive status information. In the future, we might consider monitoring WebSphere MQ FTE status information using Tivoli OMEGAMON.

For step-by-step instructions, see “Instructions for 3.11, ‘WebSphere MQ File Transfer Edition on AIX’” on page 199.

3.11.4 Debugging WebSphere MQ FTE problems

WebSphere MQ FTE agent logs hold status and error messages for agents. The agents also capture first failure data captures (FFDCs). For both logs and FFDCs, see the `<WMQ FTE data>/< coordination queue manager name>/agents/<agent name>/logs` directory. For example, see the `/var/fte/LGI.BACK.FTE.CQM/agents/AIX01/logs` directory.

WebSphere MQ FTE is a WebSphere MQ application. In the event of access control problems, consult the WebSphere MQ documentation for z/OS.

3.11.5 Application deployment

No application deployment is required for WebSphere MQ FTE on AIX.

3.12 IBM Tivoli Composite Application Manager on AIX

This section explains how to configure IBM Tivoli Composite Application Manager agents to meet the needs of the LGI scenario.

3.12.1 Configuration of the AIX agents and data collectors

The LGI scenario environment contains a WebSphere Application Server and Portal cluster that hosts the Web application. Each machine requires an IBM Tivoli Enterprise Management Agent and an IBM Tivoli Composite Application Manager for WebSphere Data Collector. The WebSphere Data Collector communicates with the Tivoli Enterprise Management Agent, which in turn communicates with the back-end Tivoli Enterprise Management Server.

On each front-end machine:

1. Before installing Tivoli Enterprise Management Agent or WebSphere Data Collector, ensure that the back-end IBM Tivoli Monitoring environment that includes the Tivoli Enterprise Management Server is installed and configured.
2. Install and configure the Tivoli Enterprise Management Agent.
3. Install and configure the WebSphere Data Collector.
4. Apply IBM Tivoli Composite Application Manager for WebSphere Data Collector iFix 12 for WebSphere Portal Server 6.1 support.

Note: If you are using IBM Tivoli Composite Application Manager for Web Resources instead of IBM Tivoli Composite Application Manager for WebSphere, use WebSphere Data Collector Fix Pack 4 for support of WebSphere Portal Server 6.1.

5. Check the configuration in the Tivoli Enterprise Portal.

For step-by-step instructions, see “Instructions for 3.12, ‘IBM Tivoli Composite Application Manager on AIX’” on page 201.

3.12.2 Application deployment

No application deployment is required for IBM Tivoli Composite Application Manager on AIX.

Deployment of the back-end components

This chapter introduces the back-end components of the Lord General Insurance (LGI) scenario. These components are used to provide data processing function in the scenario.

This chapter includes the following topics:

- ▶ “Overview of the back-end components” on page 100
- ▶ “WebSphere Application Server 6.1 on z/OS” on page 100
- ▶ “WebSphere Process Server on z/OS” on page 103
- ▶ “WebSphere Service Registry and Repository on z/OS” on page 110
- ▶ “WebSphere MQ on z/OS” on page 113
- ▶ “WebSphere Message Broker on z/OS” on page 122
- ▶ “DB2 Enterprise 9 on z/OS” on page 126
- ▶ “CICS Transaction Server v3.2 for z/OS” on page 129
- ▶ “WebSphere MQ File Transfer Edition on AIX” on page 130
- ▶ “WebSphere MQ File Transfer Edition on z/OS” on page 132
- ▶ “IBM Tivoli Monitoring on AIX” on page 134

4.1 Overview of the back-end components

The back-end components that are used for mainline application processing are deployed to a z/OS environment. These back-end components are then accessed by using the front-end components running either on AIX, Linux for System z, or z/OS. The back-end components use security features present on the z/OS platform to handle the security requirements of the scenario.

The following products are deployed on the z/OS back-end server:

- ▶ WebSphere Application Server 6.1
- ▶ WebSphere Process Server 6.1
- ▶ WebSphere Service Registry and Repository 6.1
- ▶ WebSphere MQ 6.0
- ▶ WebSphere MQ File Transfer Edition 7.0
- ▶ WebSphere Message Broker 6.1
- ▶ DB2 9
- ▶ CICS 3.2

In addition to the core z/OS back-end systems for mainline application processing, supporting products are deployed on other platforms as follows:

- ▶ WebSphere MQ 7.0 (AIX)
- ▶ IBM Tivoli Monitoring 6.2 (AIX)

4.2 WebSphere Application Server 6.1 on z/OS

The WebSphere Application Server A2 cell described in 3.1, “Front-end overview” on page 52, is used to provide access to DB2 and WebSphere MQ. This cell also provides servers for the back-end deployment of the following applications:

- ▶ Direct Car (DC)
- ▶ Credit Check
- ▶ Market Value Reduction (MVR)
- ▶ Underwriter Web Service

4.2.1 Configuration

Chapter 3, “Front-end deployment (AIX, Linux for System z, and z/OS)” on page 51, describes the overall WebSphere Application Server Network Deployment cell configuration and the clusters that are used by these applications.

DB2 Access

Access to the DB2 databases required by the Direct Car application is defined as a DB2 Universal Database Type 2 local connection by using Resource Recovery Services (RRS). The provider must be non-XA for this type of local connection.

WebSphere MQ resources

Java Message Service (JMS) queue connection factories are used to connect to the queue managers on z/OS. The connections use the bindings mode to connect locally. The *bindings mode* means the application connects to a queue manager that resides on the same system as the application. It does not use WebSphere MQ channels. If the application and queue manager reside on different systems, WebSphere MQ channels are required, and client mode is used.

4.2.2 Security configuration

Security for the back-end applications is achieved using the following profiles:

- ▶ MQ Resource Access Control Facility (RACF) profiles
- ▶ DB2 RACF profiles
- ▶ WebSphere Application Server application security and RACF ejbrole profiles

DB2 security

Access to DB2 is controlled by RACF profiles. The data source in WebSphere Application Server is defined to use a component-managed authentication alias. A component-managed alias represents a combination of a user ID and password that is specified in an application for data source authentication. On z/OS, this can be an RACF user ID that is granted access to the appropriate RACF profile used by DB2.

For additional information, see the “Using the DB2 Universal JDBC Driver to access DB2 for z/OS” topic in the WebSphere Application Server for z/OS Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tadat_jdbcd2cf2.html

WebSphere MQ security

WebSphere MQ security is provided by RACF profiles. On z/OS, the WebSphere Application Server components run as started tasks. The RACF class STARTED is used to assign user IDs to task names. The user IDs defined for the application servers must have access to the appropriate WebSphere MQ RACF profiles for the WebSphere MQ resources used by the LGI application.

The following sample RACF profile is used to associate the application server PSA2S11S with user ID PSA2SSR1:

```
RDEFINE STARTED PSA2S11S.* STDATA(USER(PSA2SSR1) GROUP(PSA2CFG)
TRACE(YES))
```

The user ID PSA2SSR1 must be granted access to the WebSphere MQ RACF profiles.

Web Services Security

The Underwriter and MVR Check Web services have been configured to use Web Services Security (WS-Security). For further details, see 5.1, “Web Services Security on AIX and z/OS” on page 136.

4.2.3 Application deployment

To deploy the LGI back-end applications, you must define the required DB2 and WebSphere MQ resources in the WebSphere Application Server Integrated Solutions Console (ISC). This section describes those tasks at a high level. For detailed information about performing these tasks, see the WebSphere Application Server Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/welcome_nd.html

We completed the following tasks to deploy the Direct Car, Credit Check, MVR, and UnderWriter applications in WebSphere Application Server:

1. Defined a DB2 Universal Database non-XA Java Database Connectivity (JDBC) provider.
2. Defined a JDBC type 2 data source under this provider.
3. Created the JMS resources required by the applications, which includes the following elements:
 - Queue Connection Factories
 - Queues
4. Installed the application enterprise archive (EAR) files.
5. Verified that the application starts successfully.

4.3 WebSphere Process Server on z/OS

This section explains how to configure WebSphere Process Server 6.1 to meet the needs of the LGI scenario.

4.3.1 Configuration

Our WebSphere Process Server configuration is set up as a Network Deployment topology with a node on each system in the z/OS Parallel Sysplex. A cluster is defined so that a server on each logical partition (LPAR) can be configured with the application and all required resources. See 3.8, “WebSphere Application Server 6.1 Network Deployment on z/OS” on page 80, for additional information about the WebSphere Application Server setup to support WebSphere Process Server.

Figure 4-1 shows the Network Deployment topology that hosts the WebSphere Process Server. The topology is configured in a manner similar to the other WebSphere Application Server cells, but only one cluster is defined.

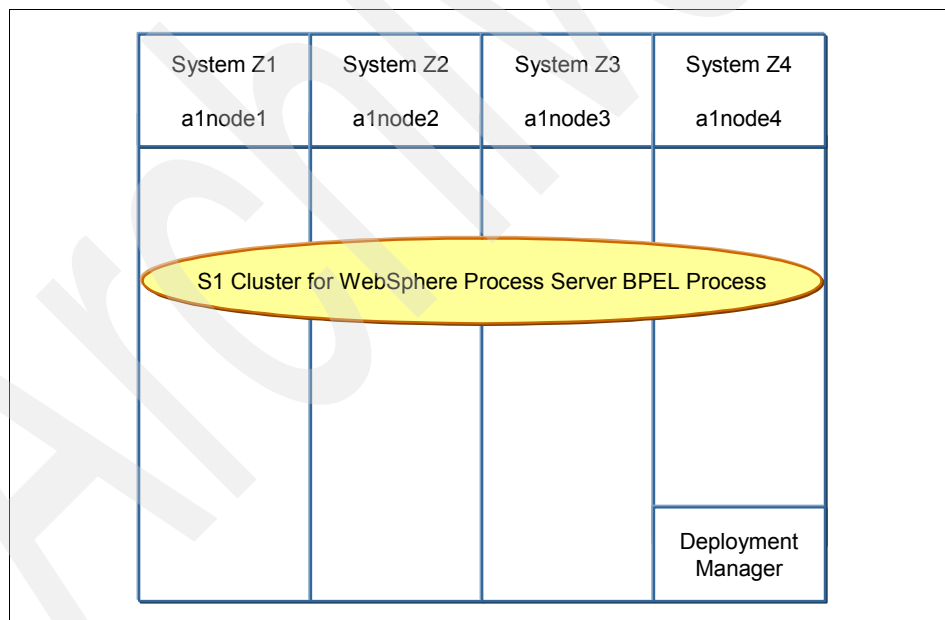


Figure 4-1 Network Deployment topology for WebSphere Process Server

Our WebSphere Process Server is at service level IBM WebSphere Process Server, 6.1.0.0, Build Number: o0748.10, Build Date: 12/5/07. All JDBC data

sources for the WebSphere Process Server databases are defined as type 2 local connections by using RRS.

4.3.2 Service integration configuration

The following service integration buses (SIBuses) are defined in our WebSphere Process Server cell:

- ▶ BPC.a1cell.Bus
- ▶ SCA.APPLICATION.a1cell.Bus
- ▶ SCA.SYSTEM.a1cell.Bus

We also have two foreign buses:

- ▶ CSQ3, a direct WebSphere MQ link to our z/OS queue manager
- ▶ SCA.SYSTEM.a1cell.Bus

The CSQ3 foreign bus uses sender/receiver channels to communicate with a WebSphere MQ queue manager that resides on a z/OS LPAR. Channels are required in WebSphere Process Server as the method you must use to connect to an external queue manager. WebSphere Application Server does not support bindings mode connections to a local queue manager. This connection is defined by creating a WebSphere MQ link for the foreign bus. Figure 4-2 provides an example of what is shown in the administrative console during this process.

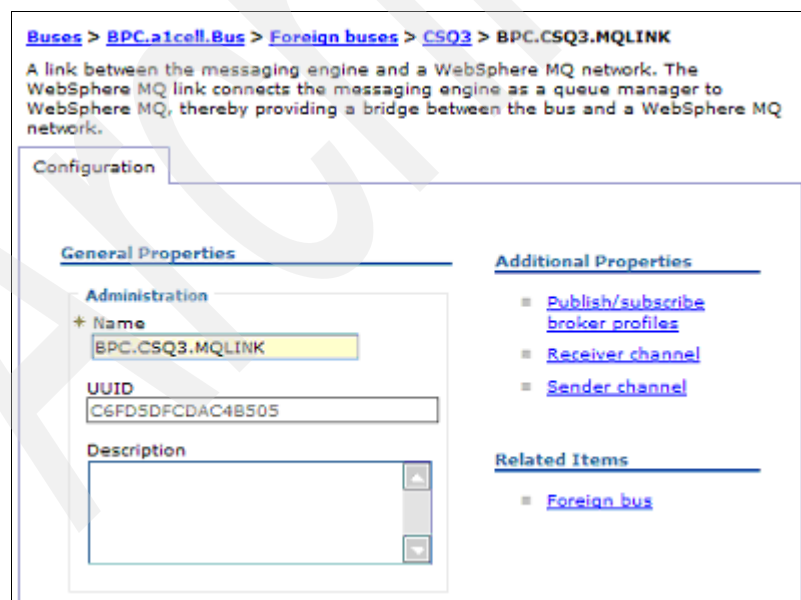


Figure 4-2 Administrative console view of a WebSphere MQ link definition

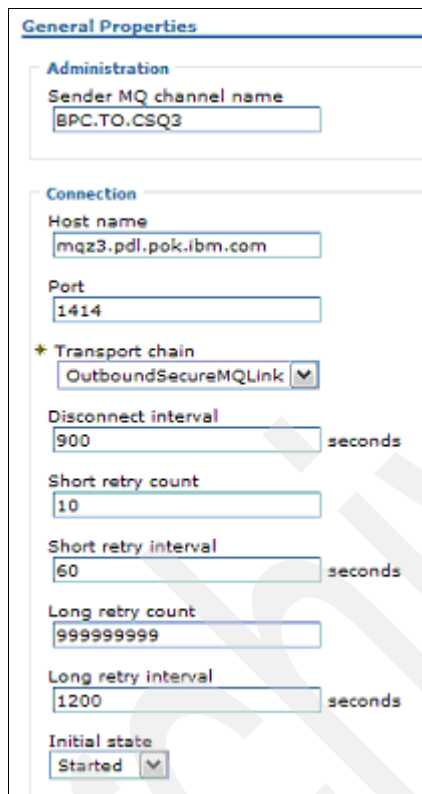
The WebSphere MQ link needs a sender and receiver channel defined to communicate with the external queue manager. Figure 4-3 shows the receiver channel that is defined to support access of the WebSphere Process Server messaging engine to the WebSphere MQ queue manager.

The screenshot shows a configuration window for a receiver channel. It is divided into two main sections: 'Connection' and 'Quality of Service'. The 'Connection' section includes fields for 'Foreign bus name' (set to CSQ3), 'Queue manager name' (set to BPC.QMGR), 'Batch size' (50), 'Maximum message size' (4194304 bytes), 'Heartbeat interval' (300 seconds), 'Sequence wrap' (99999999), and an 'Adoptable' checkbox which is checked. The 'Initial state' is set to 'Started'. The 'Quality of Service' section includes a 'Nonpersistent message speed' dropdown set to 'Fast'.

| Section | Property | Value |
|--------------------|-----------------------------|-------------------------------------|
| Connection | Foreign bus name | CSQ3 |
| | Queue manager name | BPC.QMGR |
| | Batch size | 50 |
| | Maximum message size | 4194304 bytes |
| | Heartbeat interval | 300 seconds |
| | Sequence wrap | 99999999 |
| | Adoptable | <input checked="" type="checkbox"/> |
| | Initial state | Started |
| Quality of Service | Nonpersistent message speed | Fast |

Figure 4-3 WebSphere MQ link receiver channel definition

Figure 4-4 shows the necessary details for the process of defining the WebSphere MQ link sender channel. It shows the General Properties page used for sender channel definition.



The screenshot displays the 'General Properties' configuration window for a WebSphere MQ link sender channel. It is divided into two main sections: 'Administration' and 'Connection'. In the 'Administration' section, the 'Sender MQ channel name' is set to 'BPC.TO.CSQ3'. The 'Connection' section contains several fields: 'Host name' is 'mqz3.pdl.pok.ibm.com', 'Port' is '1414', 'Transport chain' is set to 'OutboundSecureMQLink' (indicated by a plus sign and a dropdown arrow), 'Disconnect interval' is '900' seconds, 'Short retry count' is '10', 'Short retry interval' is '60' seconds, 'Long retry count' is '99999999', 'Long retry interval' is '1200' seconds, and 'Initial state' is set to 'Started' (indicated by a dropdown arrow).

| General Properties | |
|------------------------|------------------------|
| Administration | |
| Sender MQ channel name | BPC.TO.CSQ3 |
| Connection | |
| Host name | mqz3.pdl.pok.ibm.com |
| Port | 1414 |
| + Transport chain | OutboundSecureMQLink ▼ |
| Disconnect interval | 900 seconds |
| Short retry count | 10 |
| Short retry interval | 60 seconds |
| Long retry count | 99999999 |
| Long retry interval | 1200 seconds |
| Initial state | Started ▼ |

Figure 4-4 WebSphere MQ link sender channel definition

The next section explains how to secure the WebSphere MQ link between WebSphere Process Server and the WebSphere MQ queue manager.

4.3.3 Security configuration

Secure Sockets Layer (SSL) is used to secure the communications link between WebSphere Process Server and WebSphere MQ. In the following section, we explain how to secure a WebSphere MQ link to an external queue manager by using WebSphere MQ SSL channels.

Securing the WebSphere MQ link with SSL

In the LGI scenario environment, communication between WebSphere Process Server and WebSphere Message Broker uses the WebSphere MQ Link. The communication over the WebSphere MQ Link is secured by using SSL (Figure 4-5).

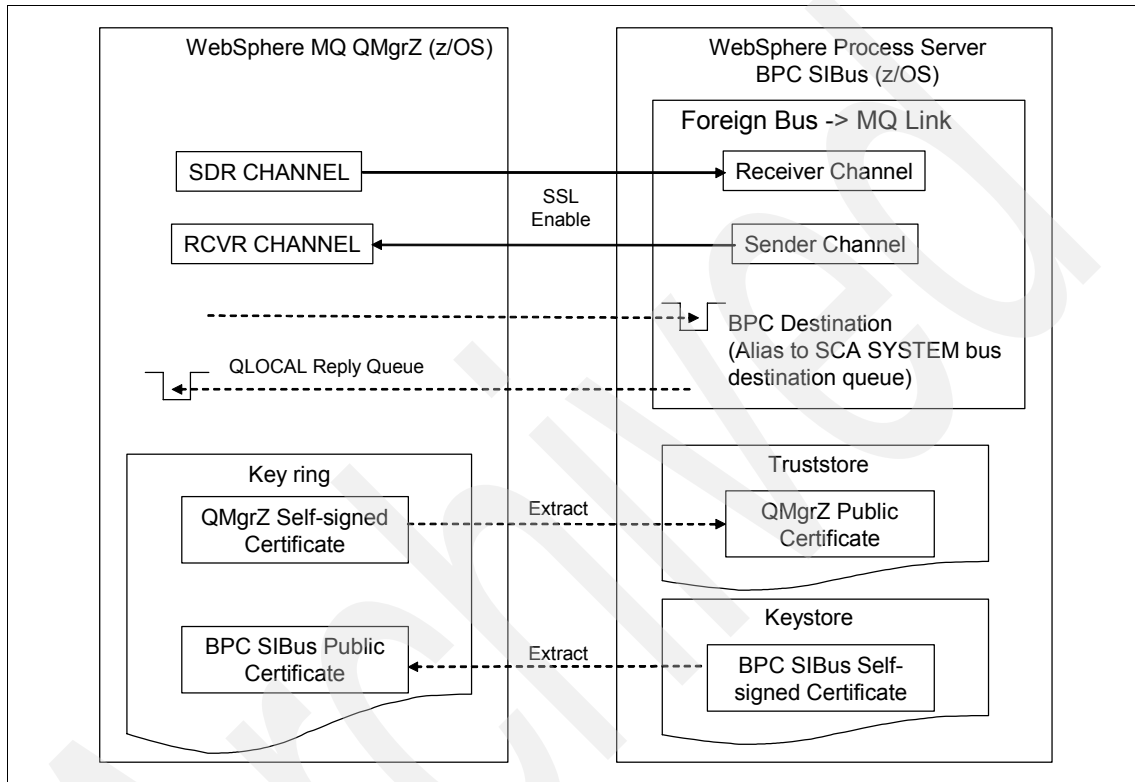


Figure 4-5 WebSphere Process Server to WebSphere MQ by using an MQ link SSL architecture

To configure SSL on the WebSphere MQ link sender and receiver channels:

1. Create a keystore for the business process container SIBus.
2. Create a self-signed certificate for the business process container SIBus.
3. Extract the certificate for the business process container SIBus.
4. Create a truststore for the business process container SIBus.
5. Add the WebSphere MQ queue manager's public certificate to the truststore.
6. Create an SSL configuration for the business process container SIBus.
7. Associate the SSL configuration with the inbound WebSphere MQ link.
8. Associate the SSL configuration with the outbound WebSphere MQ link.
9. Configure the WebSphere MQ link sender channel to use SSL.
10. Enable security on the business process container SIBus.

11. Define users who are authorized to exchange messages between SIBuses.
12. Stop and start the WebSphere MQ link channels for changes to take effect.

For step-by-step instructions associated with these steps, see “Instructions for 4.3, ‘WebSphere Process Server on z/OS’” on page 204. For instructions on how to configure the WebSphere MQ queue manager end of these SSL channels, see 4.5, “WebSphere MQ on z/OS” on page 113.

Exchanging certificates on z/OS: When exchanging certificates on z/OS, use the binary DER format and FTP binary mode to avoid any certificate corruption that might be caused by conversion problems between ASCII and EBDIC. Use WebSphere MQ SupportPac MO04 for further tips on sending SSL certificates between platforms by using FTP.

Web Services Security

The business process flow that acts as a Web service client to the MVR Check Web services has been configured to use WS-Security. For further details, see 5.1, “Web Services Security on AIX and z/OS” on page 136.

4.3.4 WebSphere Process Server security

Since the WebSphere Application Server cells are set up to use RACF to manage authentication, ejbrole definitions are necessary to allow access to the WebSphere Process Server components. We defined the following ejbrole profiles:

- ▶ BPEAPIUser
- ▶ BPESystemAdministrator
- ▶ BPESystemMonitor
- ▶ WebClientUser
- ▶ JMSAPIUser
- ▶ TaskAPIUser
- ▶ TaskSystemAdministrator
- ▶ TaskSystemMonitor
- ▶ EscalationUser
- ▶ WBIOperator

4.3.5 Application deployment

The LGI application has one Business Process Execution Language (BPEL) process installed into WebSphere Process Server. Figure 4-6 shows the module components.



Figure 4-6 WebSphere Process Server BPEL process module list

Queue connection factories are associated with the SCA.SYSTEM.a1cell.Bus. Access to the external queue manager is through a foreign bus, CSQ3, that is defined as an indirect link by using the business process container bus. Figure 4-7 illustrates the connections from the SCA system bus that is referenced by the JMS Queue Connection Factory and the CSQ3 external queue manager. WebSphere MQ access is achieved by using the business process container foreign bus indirect link to CSQ3 as shown in Figure 4-7.

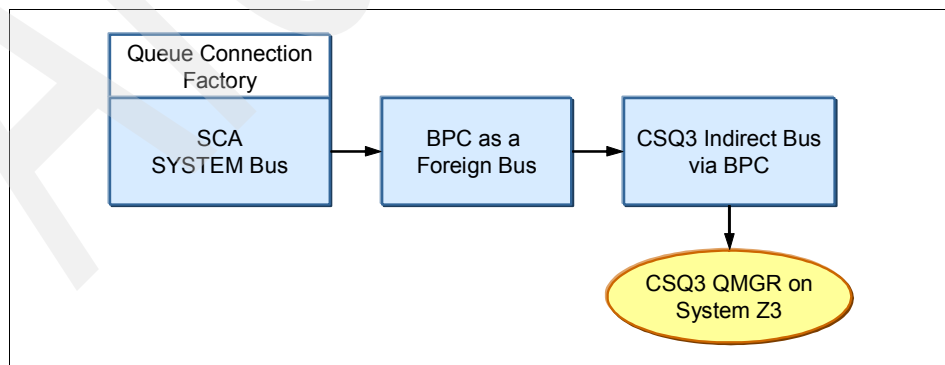


Figure 4-7 Bus link topology

4.4 WebSphere Service Registry and Repository on z/OS

WebSphere Service Registry and Repository is set up in the WebSphere Application Server A3 cell. It is set up in a topology similar to WebSphere Process Server. See 3.1, “Front-end overview” on page 52, for additional information.

4.4.1 Configuration

Figure 4-8 shows our WebSphere Application Server cluster for WebSphere Service Registry and Repository. It illustrates how one cluster is spread across all the LPARs in the Parallel Sysplex. Each cluster contains one node and server. The WebSphere Service Registry and Repository application is installed at the cluster level for availability on any system in the sysplex.



Figure 4-8 WebSphere Application Server cluster for WebSphere Service Registry and Repository

There are no requirements for JMS resources. JDBC resources are those required by the WebSphere Service Registry and Repository application and are defined automatically by the WebSphere Service Registry and Repository installation scripts.

4.4.2 Security configuration

Administrative security for the Network Deployment environment is set up in the same manner in which we set up administrative security for the other cells that support the application and WebSphere Process Server. Additional administrative security is used to secure the WebSphere Service Registry and Repository application itself.

WebSphere Service Registry and Repository administrative security

WebSphere Service Registry and Repository consists of applications that are installed into a WebSphere Application Server cell. The applications are ServiceRegistry and ServiceRegistryTS. Since WebSphere Service Registry and Repository is an application, security is accomplished by enabling application security in WebSphere Application Server and then defining the appropriate RACF ejbrole profiles required by WebSphere Service Registry and Repository. This includes the following profiles:

- ▶ Administrator
- ▶ User

WebSphere Application Server has an ejbrole “administrator” role with a lowercase letter “a.” The WebSphere Service Registry and Repository Administrator ejbrole is distinguished by the uppercase letter “A.”

WebSphere Service Registry and Repository application security

WebSphere Service Registry and Repository application security requires SSL to be configured for clients, such as WebSphere Message Broker, to access WebSphere Service Registry and Repository services. This is accomplished by performing the following tasks:

- ▶ Extract the default certificate in binary from WebSphere Application Server to a file in z/OS UNIX System Services.
- ▶ Create a jks keystore for the WebSphere Message Broker to use.
- ▶ Import the certificate into the jks keystore.

For step-by-step instructions on how to extract the certificate in WebSphere Application Server, see “Instructions for 4.4, ‘WebSphere Service Registry and Repository on z/OS’” on page 210.

Keytool can be used to import the certificate into a truststore used by WebSphere Message Broker. For further explanation, see “Configuring a broker

to access a secure WebSphere Service Registry and Repository server” on page 123.

4.4.3 Application deployment

The LGI application uses WebSphere Service Registry and Repository to store the following information:

- ▶ The RiskAdjustment Web Service Definition Language (WSDL) file and provide access to this Web service upon request
- ▶ WSDL files that represent WebSphere MQ endpoints

Developing the RiskAdjustment WSDL file

The WSDL file for the RiskAdjustment Web service was auto-generated from a Java bean by using IBM Rational Application Developer (RAD).

Developing the SOA Messaging WSDL file

The WSDL file is manually created to allow a port type to resolve to a WebSphere MQ Internationalized Resource Identifier (IRI). For more information about WebSphere MQ IRIs, see WebSphere MQ SupportPac MA93.

It is possible to store each WebSphere MQ endpoint in a separate WSDL document or to store all WebSphere MQ endpoints in a common WSDL document. The endpoints are used by WebSphere Message Broker message flows. Therefore, the WSDL documents contain endpoints grouped by WebSphere Message Broker message flow. For more information, see “Instructions for 4.4, ‘WebSphere Service Registry and Repository on z/OS’” on page 210.

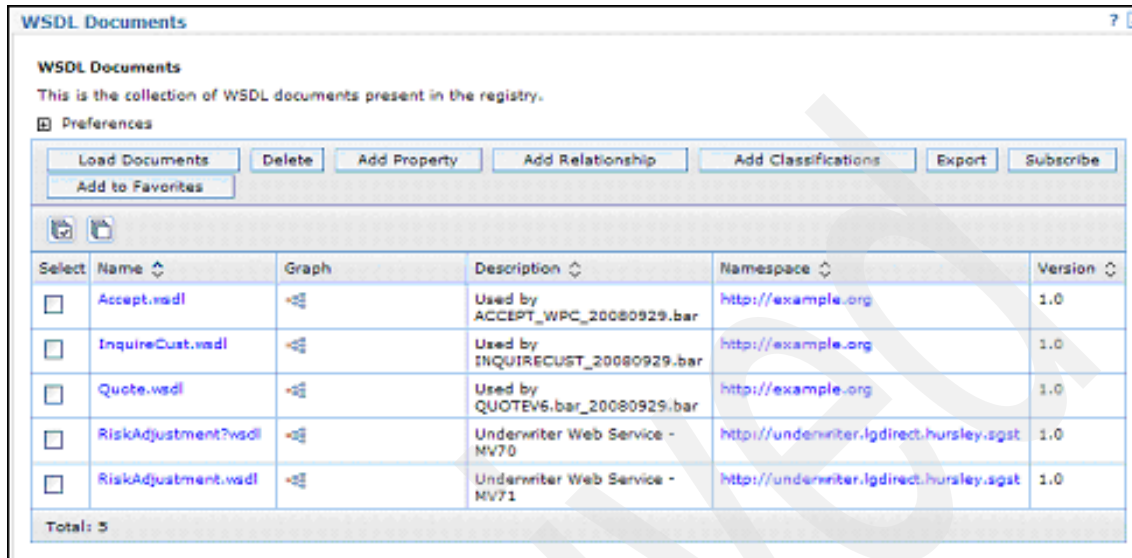
At this time in the LGI scenario, the WSDL is only used for endpoint routing. Therefore, creating abstract portions of the WSDL to describe message types is not required. This helps to simplify the WSDL. Despite this simplification, for users new to WSDL concepts, manual creation of WSDL can be difficult. Tooling can help reduce difficulties in this area. WebSphere MQ 7.0.0.1 Explorer offers a WSDL creation plug-in, which provides a simple GUI for creating WSDL (representing WebSphere MQ endpoints).

Installing a WSDL document into WebSphere Service Registry and Repository

To load the WSDL files into WebSphere Service Registry and Repository:

1. Log in to the WebSphere Service Registry and Repository application.
2. In Service Documents, select **Load Documents**.

3. Complete the fields with the appropriate WSDL file name and other required information. Figure 4-9 shows an example of the final results.



WSDL Documents

This is the collection of WSDL documents present in the registry.

Preferences

Load Documents Delete Add Property Add Relationship Add Classifications Export Subscribe

Add to Favorites

| Select | Name | Graph | Description | Namespace | Version |
|--------------------------|---------------------|--|----------------------------------|--|---------|
| <input type="checkbox"/> | Accept.wsdl | | Used by ACCEPT_WPC_20080929.bar | http://example.org | 1.0 |
| <input type="checkbox"/> | InquireCust.wsdl | | Used by INQUIRECUST_20080929.bar | http://example.org | 1.0 |
| <input type="checkbox"/> | Quote.wsdl | | Used by QUOTEV6.bar_20080929.bar | http://example.org | 1.0 |
| <input type="checkbox"/> | RiskAdjustment?wsdl | | Underwriter Web Service - MV70 | http://underwriter.lgdirect.hursley.sgst | 1.0 |
| <input type="checkbox"/> | RiskAdjustment.wsdl | | Underwriter Web Service - MV71 | http://underwriter.lgdirect.hursley.sgst | 1.0 |

Total: 5

Figure 4-9 WSDL document in the WebSphere Service Registry and Repository console

Note: Each piece of WSDL holding information related to WebSphere MQ resources is associated with a WebSphere Message Broker message flow. Therefore, for ease of use, the description is set to the name of the broker archive (BAR) file that contains the message flows.

4.5 WebSphere MQ on z/OS

This section describes how to configure WebSphere MQ objects to function within the LGI scenario.

4.5.1 Configuration

The LGI scenario environment consists of several back-end queue managers, including the following types:

- ▶ Those that are local to the message brokers, one queue manager per broker
- ▶ Those that are local to the LGI back-end CICS regions, one queue manager per CICS region

- ▶ Those that are local to the DC back-end server for WebSphere Application Server, one queue manager per server (These servers are on AIX in the mixed stack scenario implementation.)

All queue managers except for those that are local to the DC back-end server for WebSphere Application Server are configured to be part of a single WebSphere MQ cluster.

Where there are multiple queue managers performing the same role (for example, the broker queue managers), queue-sharing is configured to allow for workload balancing and failover.

To install and configure WebSphere MQ on z/OS:

1. Install WebSphere MQ as explained in “Installing WebSphere MQ for z/OS” from the WebSphere MQ 6.0 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqsat.doc/csq82du.htm>
2. Create and customize the queue managers, channel initiators, and queue-sharing group. Add them to the queue-sharing group created in Chapter 3, “Front-end deployment (AIX, Linux for System z, and z/OS)” on page 51. Follow the instructions in “Customizing your queue managers,” in the WebSphere MQ 6.0 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqsav.doc/csqsav0410.htm>
3. Configure the MQ-CICS bridge by following the instructions in the “Customizing for CICS” topic of the WebSphere MQ 6.0 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqsav.doc/csq83hx.htm>
4. Add the queue manager to the existing WebSphere MQ cluster that you created in 3.1, “Front-end overview” on page 52.
5. Define the queues that are required by the application to the queue manager. Define the request queues (both for the Web application and the MQ-CICS Bridge) as shared queues for either of the following situations:
 - More than one queue manager is capable of handling the request
 - Shared queues enable faster communication between the systems making the requests and those handling the requests.

CICS-WebSphere MQ adapter: The CICS-WebSphere MQ adapter can be used in place of the WebSphere MQ-CICS bridge where changes to the CICS application are acceptable. For more information, see “The CICS-WebSphere MQ adapter” topic of the CICS Transaction Server for z/OS, Version 3.2 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/cicsts/v3r2/topic/com.ibm.cics.ts.wmq.adapter.doc/topics/zc12060_.html

4.5.2 Security configuration

The following sections describe the configuration changes that we made to enable security in WebSphere MQ that are specific to the LGI project.

Configuring access control

To provide sufficient access to the message brokers and CICS regions to the queue manager resources that they need for the LGI Web application:

1. Log on to a Time Sharing Option (TSO) session with a user ID that is part of a RACF group that has administrative authorities.
2. Define the RACF profiles at the queue-sharing group level and grant update access to all local request queues for the group, which allows the brokers to obtain messages.

Note the following additional considerations for this step:

- All other queues that are only used by the message broker message flows use the same permissions as indicated previously.
 - If the request queue is shared and accessible by the queue managers local to the WebSphere Application Server, you also need additional permissions.
3. Define the RACF profiles and grant the permissions required to the CICS regions for the WebSphere MQ-CICS Bridge initiation queue and request.
 4. Grant permission for the reply queue specified by the message broker.
 5. Refresh the queue manager’s security configuration to implement the changes.
 6. Log off the WebSphere MQ Administrator.

In addition to securing local access control, secure each queue manager for remote access. Any channel without an MCAUSER value allows user impersonation and anonymous administrative authority. The LGI front-end and back-end queue managers do not share a single security domain. Therefore,

using context channel security (by setting the PUTAUT channel attribute to CTX) can be difficult to maintain. As a result, default security channel security is used (by setting the PUTAUT channel attribute to DEF) in conjunction with MCAUSER (set to a user with low-privileged group).

For more detailed instructions associated with these steps, see “Instructions for 4.5, ‘WebSphere MQ on z/OS’” on page 213.

Configuring SSL

The LGI scenario environment involves configuring SSL on the following channels:

- ▶ The WebSphere MQ cluster sender and receiver channels between the z/OS queue managers used by the message brokers and the queue managers used by the WebSphere Application Server or Portal servers hosting the Web application on AIX or Linux for System z

Client authentication is enabled to prevent queue managers without a certificate from connecting to the cluster.

For instructions to configure SSL on the AIX or Linux for System z queue manager end of these channels, see “Instructions for 3.5, ‘WebSphere MQ on AIX and Linux for System z’” on page 183. For instructions to configure SSL between the z/OS queue managers, see “Instructions for 3.9, ‘WebSphere MQ on z/OS’” on page 195.

- ▶ The sender and receiver channels between the z/OS queue managers used by the message brokers and the WebSphere MQ link defined on the SIBus of the WebSphere Process Server

For instructions to configure SSL on the WebSphere MQ link on the WebSphere Process Server end of these channels, see “WebSphere Process Server on z/OS” on page 103.

Configuring a z/OS queue manager to use SSL

To enable a z/OS queue manager to use SSL and create a key ring and self-signed certificate for the queue manager:

1. Grant the channel initiator address space user ID access to RACF key rings.
2. Enable the queue managers to use SSL by setting the queue manager's property SSL tasks (SSLTASKS). The value must be greater than 1 but less than 50 to prevent storage allocation problems.
3. Restart the channel initiator address space (CHINIT) to implement the change.
4. Create the key ring for the queue manager.

5. Add the key ring to the queue manager by setting the queue manager's property SSL key repository (SSLKEYR).
6. Create a self-signed certificate for the local queue manager.
7. Add the self-signed certificate to queue manager's key ring.
8. Export the local queue manager's public certificate so that it is available to be added later into the remote queue managers' key rings and key databases.
9. Refresh the queue manager's SSL settings for the changes to take effect.

Note the following considerations regarding these steps:

- ▶ This scenario does not use the Integrated Cryptographic Service Facility (ICSF) to store certificates. Extra setup steps are required if you want to use ICSF.
- ▶ In this example, RACF commands are issued from the TSO command interface, while being logged on as a user with sufficient RACF authority to create key rings and certificates. In addition, the MQM commands are issued from the Interactive System Productivity Facility (ISPF) MQ utilities interface.

For more detailed instructions regarding these steps, see “Instructions for 4.5, ‘WebSphere MQ on z/OS’” on page 213.

Exchanging certificates: AIX and Linux for System z queue manager

The following steps are required on a z/OS queue manager to enable SSL on sender and receiver cluster channels to and from the remote queue manager. For the contents in this section, the remote queue managers can be either on AIX or Linux for System z. The behavior is the same unless otherwise specified.

Figure 4-10 shows the SSL configuration between the remote and z/OS queue managers.

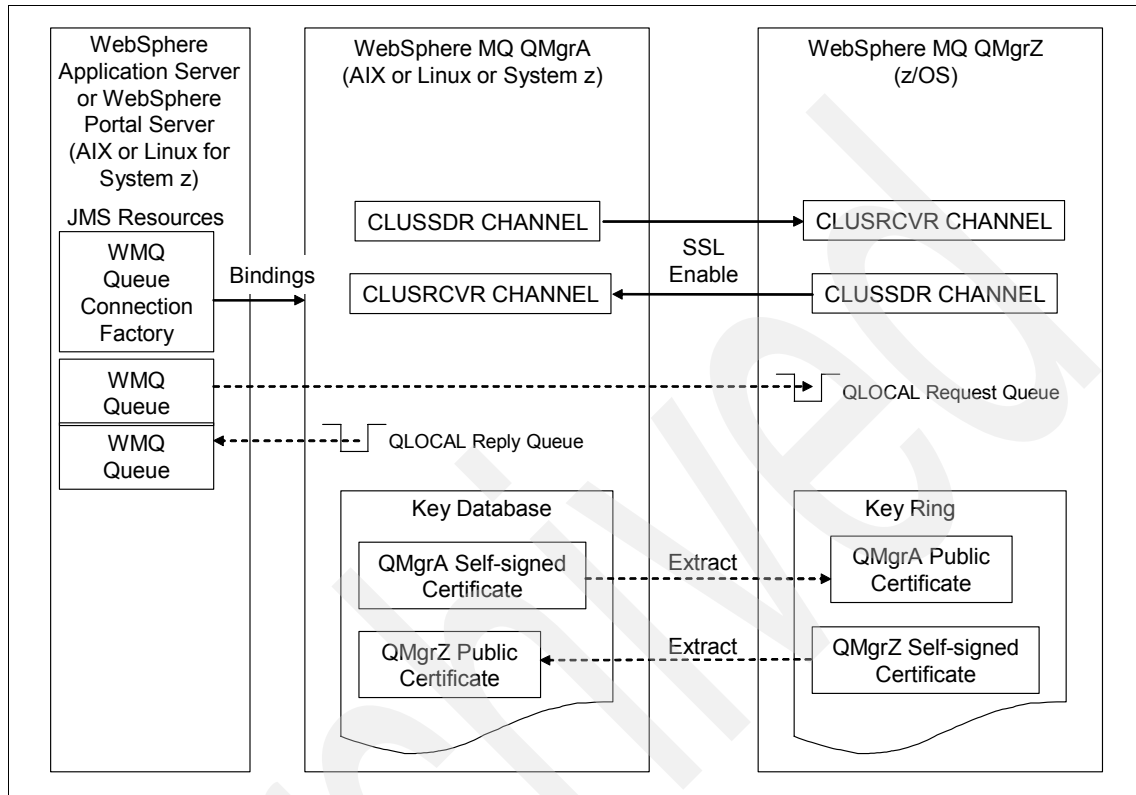


Figure 4-10 SSL configuration between remote and z/OS queue managers

In the following steps, we issue RACF commands from the ISPF TSO command interface while logged on as a user with sufficient RACF authority to create key rings and certificates. We issue the MQM commands from the ISPF MQ utilities interface.

To enable SSL on a z/OS queue manager's cluster sender and receiver channels to a remote queue manager:

1. Configure the z/OS queue manager to use SSL by following the instructions in "Configuring a z/OS queue manager to use SSL" on page 116.
2. Obtain the public certificate for the remote queue manager, for example as a Base64-encoded ASCII data certificate.
3. Use ISPF dataset utilities to precreate a z/OS sequential file into which remote public certificate will be copied.

Error message: Our attempts to add a certificate to a key ring from a sequential file created dynamically during an FTP session from AIX resulted in the error message: “An error occurred with the specified input data set.”

4. Send the file by FTP in ASCII mode from the remote system's temporary directory to the precreated z/OS sequential file.
5. Add the remote queue manager's public certificate to a local queue manager's key ring.

Informational message: When adding certificates from AIX queue managers RACF displays, we encountered the informational message “Certificate Authority not defined to RACF. Certificate added with TRUST status.”

6. For the changes to take effect, refresh the queue manager's SSL settings.
7. Set the cipher specification (SSLCIPH) on the sender channel to match the value specified on the corresponding receiver channel on the remote queue manager.
8. Set the cipher specification (SSLCIPH) to be used on the receiver channel and request client authentication.
9. Restart both ends of each cluster sender and receiver channel for the SSL settings to take effect. This includes restarting the dynamic cluster sender channels. The dynamic cluster sender channels should automatically inherit the same SSLCIPH value as the corresponding receiver channel after they are restarted.

Although these steps detail how to set up SSL certificates for communication between two queue managers, in practice, when using self-signed certificates, each queue manager must share its public certificate with every other queue manager in the cluster. This results in complex SSL certificate administration processes. Therefore, use certificate authorities in WebSphere MQ cluster environments.

For more detailed instructions associated with these steps, see “Instructions for 4.5, ‘WebSphere MQ on z/OS’” on page 213.

Exchanging certificates with a WebSphere Process Server WebSphere MQ link

The following example shows the steps that are required on a z/OS queue manager to enable SSL on sender and receiver channels to a SIBus on WebSphere Process Server by using a WebSphere MQ link. These steps apply to SIBus in any WebSphere Application Server v6.1 based product.

Figure 4-11 shows the SSL configuration between the z/OS WebSphere MQ queue managers and the z/OS SIBus.

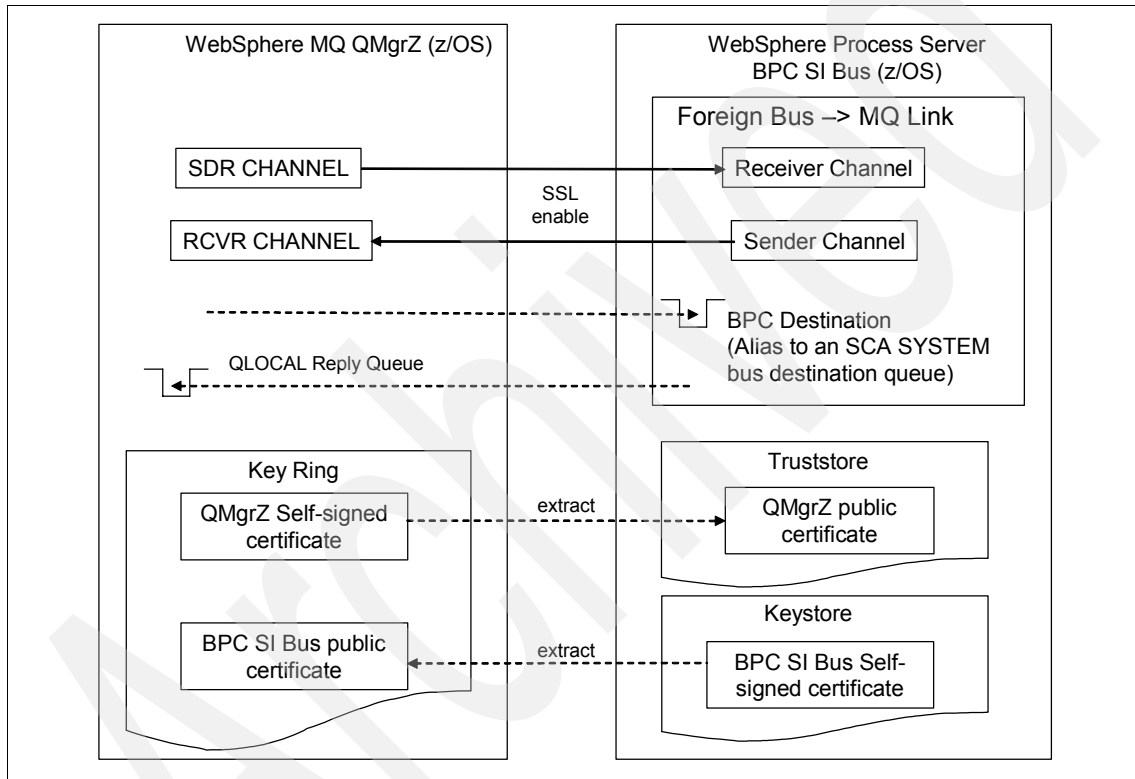


Figure 4-11 SSL configuration between queue manager and SIBus

In the following steps, we issue RACF commands from the ISPF TSO command interface while logged on as a user with sufficient RACF authority to create key rings and certificates. We issue MQM commands from the ISPF MQ utilities interface.

To enable SSL on a z/OS queue manager's sender and receiver channels to WebSphere MQ link channels on a WebSphere Process Server SIBus:

1. Configure the z/OS queue manager to use SSL by following the instructions in "Configuring a z/OS queue manager to use SSL" on page 116.
2. Obtain the public certificate for the remote WebSphere Process Server WebSphere MQ link, for example as a Binary DER data certificate.
3. Use ISPF dataset utilities to precreate a z/OS sequential file into which a remote public certificate is copied.

Error message: Our attempts to add a certificate to a key ring from a sequential file, created dynamically during an FTP session, resulted in the error message "An error occurred with the specified input data set."

4. Send the file by FTP in binary mode from the z/OS UNIX System Services temporary directory into the precreated sequential file.
5. Add the remote WebSphere MQ link (SIBus) public certificate to the local queue manager's key ring.
6. For the changes to take effect, refresh the queue manager's SSL settings.
7. Set the cipher specification (SSLCIPH) on the sender channel to match the value specified on the corresponding WebSphere MQ link receiver channel.
8. Set the cipher specification (SSLCIPH) to be used on the receiver channel, and request client authentication.
9. Restart both ends of each sender and receiver channel for the SSL settings to take effect.

For more information about these steps, see "Instructions for 4.5, 'WebSphere MQ on z/OS'" on page 213.

4.5.3 Application deployment

No application deployment is required for WebSphere MQ on z/OS.

4.6 WebSphere Message Broker on z/OS

This section explains how to configure WebSphere Message Broker to function within the LGI scenario.

4.6.1 Configuration

The LGI scenario environment involves multiple message brokers configured as part of a domain. The environment also uses WebSphere MQ queue sharing groups to provide workload balancing and high availability. The Message Broker Toolkit has been installed on a Windows machine to design the message flows. The Configuration Manager component can be installed and configured either on the same Windows machine as the toolkit or on z/OS.

To create a connection to the z/OS Configuration Manager from your toolkit component, facilitating administration to the z/OS message brokers and deployment of the message flows:

1. Install the Message Broker Toolkit on a Windows machine by following the instructions in Chapter 10, “Installing the Message Broker Toolkit,” of the WebSphere Message Broker 6.1 Installation Guide at the following address:

<http://publib.boulder.ibm.com/epubs/pdf/c3468661.pdf>

2. Install the WebSphere Message Broker code by following the instructions in the Program Directory.

To use the WebSphere Message Broker function to look up Web services in WebSphere Service Registry and Repository, ensure that Message Broker code is at minimum level of 6.1 Fix Pack 2.

3. Create a message broker by following the instructions in the “Creating a Broker on z/OS” topic in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/topic/com.ibm.etools.mft.doc/ae22400_.htm

4. Install the Message Broker Configuration Manager on the same Windows machine as the Message Broker Toolkit, following instructions in “Installing the Runtime Components” of Chapter 9 of the *WebSphere Message Broker V6.1 Installation Guide*.

5. Configure the message broker to connect to the WebSphere Service Registry and Repository server by following the instructions in the “WebSphere Service Registry and Repository” topic in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/topic/com.ibm.etools.mft.doc/ac56060_.htm

Note: A broker can only connect to a single WebSphere Service Registry and Repository server.

4.6.2 Security configuration

In order for a message broker to work successfully, some z/OS security configuration is required. For more information, see “Setting up z/OS security” in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/topic/com.ibm.etools.mft.doc/ae14030_.htm

Additionally, the LGI scenario environment secures the connection between the message broker and WebSphere Service Registry and Repository server. Fix Pack 2 must be applied to WebSphere Message Broker 6.1 to successfully use a secure connection to WebSphere Service Registry and Repository.

Configuring a broker to access a secure WebSphere Service Registry and Repository server

In the LGI scenario environment, the WebSphere Service Registry and Repository server is configured to use administration and application security. Calls from WebSphere Message Broker must flow over a secure HTTP (HTTPS) connection and must send a user ID and password to be authenticated by the WebSphere Service Registry and Repository server.

The following steps illustrate how to allow a message broker to access a secure WebSphere Service Registry and Repository server. The example is based on the example instructions in the “Accessing a secure WebSphere Service Registry and Repository” topic in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac56150_.htm

1. Obtain the WebSphere Service Registry and Repository server’s HTTP public certificate. It is assumed that administration and application security are already enabled on the WebSphere Service Registry and Repository cluster

and custom certificates have been created for use by connections by using its WC_defaulthost_secure port.

2. Create a keystore for the message broker by using a key management tool.
3. Create a truststore for the message broker by using a key management tool.
4. Import the WebSphere Service Registry and Repository certificate into the broker's truststore.
5. Modify the broker's configuration to set the URL of the WebSphere Service Registry and Repository server's HTTPS port by setting the endpointAddress value for the broker's DefaultWSRR ServiceRepository.
6. Modify the broker's configuration to access its keystore and truststore by setting the brokerKeystoreFile and brokerTruststoreFile values for the BrokerRegistry.
7. Stop the broker. The broker must be stopped to run **mqsisetdbparms** commands. The broker must also be restarted before the changes that are made through the **mqsichangeproperties** command can take effect.
8. Set the user ID and password used by the broker to access the WebSphere Service Registry and Repository server by issuing the **mqsisetdbparms** command to edit these values for the DefaultWSRR::WSRR data source.
9. Set the user ID and password used by the broker to access its keystore and truststore by issuing the **mqsisetdbparms** command to edit these values for the brokerKeystore and brokerTruststore data sources.

The user ID value is not used to access the brokerKeystore or brokerTruststore. However, the command syntax requires a value. Therefore, for the purposes of our test activities, we used *dummy*.

10. Configure the user ID and password that is used by the broker to access the JMS resources in WebSphere Service Registry and Repository used by Cache Notification.

enableCacheNotification=true: If the broker's configuration property enableCacheNotification is set to *true*, an extra step is required. This step is documented in the "Setting up Cache Notification" topic in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac56080_.htm

11. Start the broker.

12. Confirm the broker's property values by issuing the `mqsireportproperties` command to display the BrokerRegistry and DefaultWSRR ServiceRegistry property values.

For more detailed step-by-step instructions, see “Instructions for 4.6, ‘WebSphere Message Broker on z/OS’” on page 222.

Web Services Security

The message flow that is used to obtain insurance quotes, which acts as a Web service consumer (client) to the Underwriter Web services, has been configured to use WS-Security. For further details, see 5.1, “Web Services Security on AIX and z/OS” on page 136.

4.6.3 SOA messaging flow development

Where there is a requirement to route messages to queues based on information stored in the WebSphere Service Registry and Repository, the following nodes are connected in a series:

1. The *endpoint lookup node* uses the port type specified on the node to call WebSphere Service Registry and Repository for the endpoint IRI.
2. The *compute node* parses the WebSphere MQ IRI that is obtained by the endpoint lookup node to find the WebSphere MQ queue name. It sets the queue name in the destination list for use by the subsequent MQOutput node.
3. The *MQOutput node* uses the queue specified in the destination list as the output queue.

4.6.4 Application deployment

By using the Message Broker Toolkit, deploy the BAR files to the message brokers:

1. Import the BAR files from the application development version to the production version of Message Broker Toolkit.
2. Click the BAR file's **Configuration** tab and perform any customization that is required for the target environment (for example, host names, database locations, and so on).
3. Deploy the BAR files to the appropriate execution group of each of the brokers.

4.7 DB2 Enterprise 9 on z/OS

This section describes the DB2 environment and configurations that are used to support the back-end processing of the LGI and DC systems.

4.7.1 DB2 data sharing

The database design and DB2 for z/OS was chosen for this application to provide high availability. DB2 data sharing gives us that availability. A four-member data sharing group was used, and each member is assigned an LPAR within the Parallel Sysplex. This provides a clustered architecture where any transaction can be executed on any of the available DB2 members.

For this example, the application is deployed in “Plex2” as shown in Figure 4-12. The configuration shows all the LPARs configured on each System z machine. We used four LPARs (Z1, Z2, Z3, and Z4).

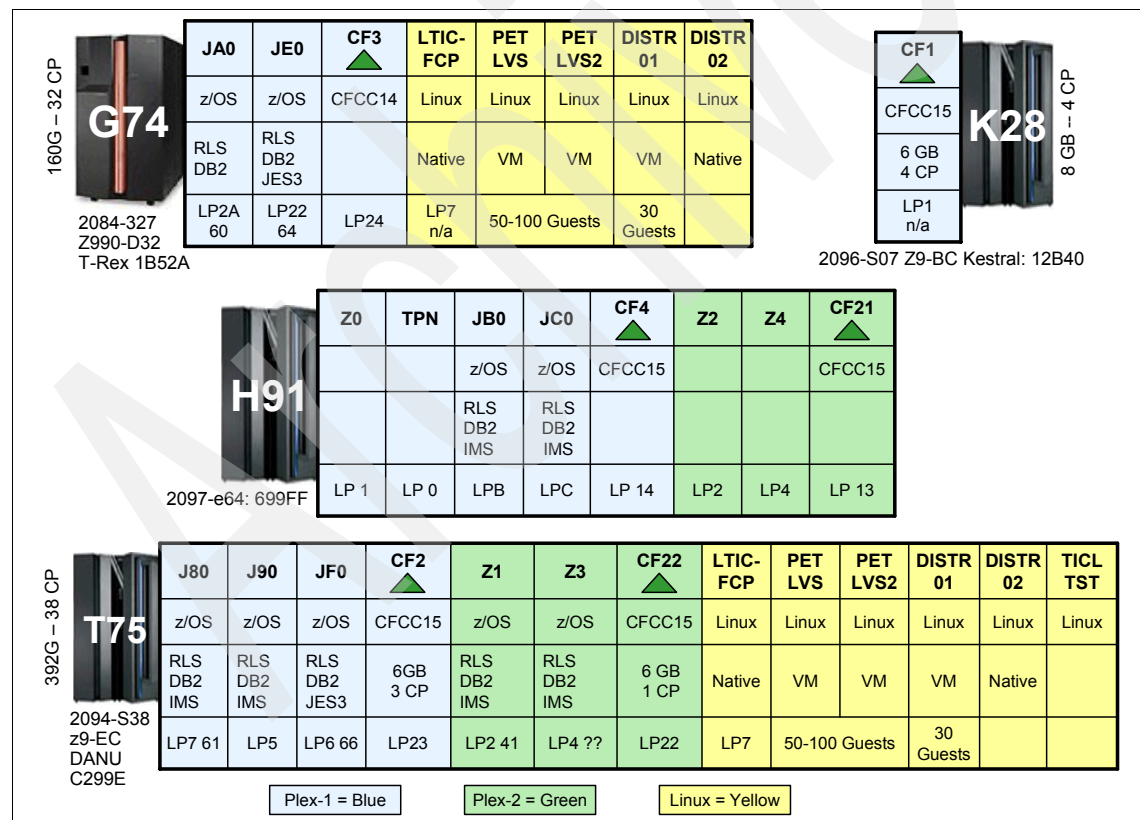


Figure 4-12 Test Parallel Sysplex configuration

Figure 4-13 illustrates the basic configuration for the DBXG data sharing group. It is a four-way data sharing group with members DBX1, DBX2, DBX3, and DBX4. The LGI and DC databases are in this group.

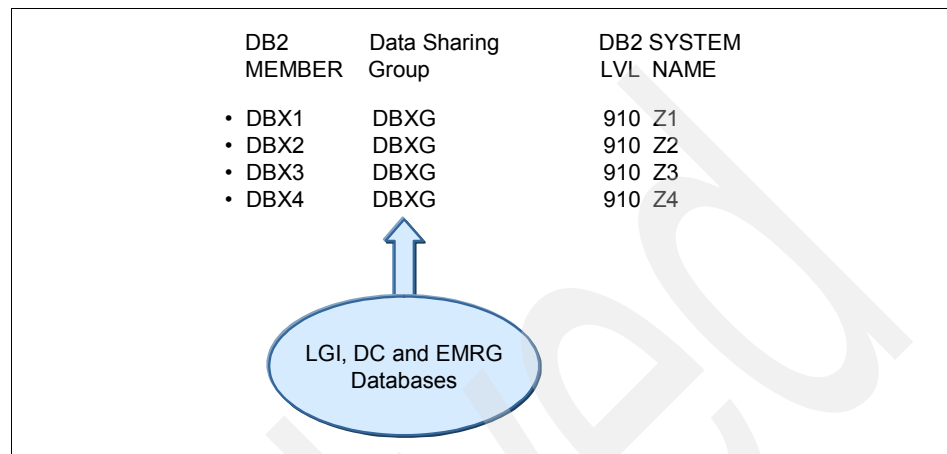


Figure 4-13 Data sharing group used for LGI

Alternative: The CICS-WebSphere MQ adapter can be used in place of the WebSphere MQ-CICS bridge where changes to the CICS application are acceptable. For further information, see the “The CICS-WebSphere MQ adapter” topic in the CICS Transaction Server for z/OS Version 3 Release 2 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/cicsts/v3r2/topic/com.ibm.cics.ts.wmq.adapter.doc/topics/zc12060_.html

Security configuration

The authentication and authorization for the LGI scenario application are done at the front end from the WebSphere components. A qualified or trusted user ID is passed to WebSphere MQ, WebSphere Message Broker, CICS, and finally DB2. The DB2 data is secured by using standard RACF controls and profiles with that user ID.

Database definitions

The following tables describe the LGI, DC, and EMRG databases.

Table 4-1 describes the database for the LGI non-secure application configuration. It provides a high level description of the LGQADB9 database that is used by the LGI back-end application components.

Table 4-1 LGQADB9 database

| Database | Tablespace | Tables | Indexes | SG | BP |
|----------|------------|------------------|-------------------------------------|--------|--------|
| LGQADB9 | LGQATS01 | LGINSUR.customer | LGINSUR.iCustomer | LGQASG | BP9(T) |
| | LGQATS02 | LGINSUR.policy | LGINSUR.iPolicy LGINSUR.iPolicy2 | | |
| | LGQATS03 | LGINSUR.motor | LGINSUR.iMotor | | |

Table 4-2 describes the database for the DC application configuration. It provides a high level description of the DCQADB9 database that is used by the DC back-end application components.

Table 4-2 DCQADB9 database

| Database | Tablespace | Tables | Indexes | SG | BP |
|----------|------------|------------------|-------------------------------------|--------|---------|
| DCQADB9 | DCQATS01 | DCINSUR.customer | DCINSUR.iCustomer | DCQASG | BP14(T) |
| | DCQATS02 | DCINSUR.policy | DCINSUR.iPolicy DCINSUR.iPolicy2 | | |
| | DCQATS03 | DCINSUR.motor | DCINSUR.iMotor | | |
| | DCQATS04 | DCINSUR.quote | DCINSUR.iQuote | | |

Table 4-3 describes the database for the EMRG application configuration. It provides a high level description of the EMRGDB9 database that is used by the message broker quote/accept back-end application components.

Table 4-3 EMRGDB9 database

| Database | Tablespace | Tables | Indexes | SG | BP |
|----------|------------|---|------------------|-----------|------------------|
| EMRGDB9 | EMRGTS01 | EMERGE.extcheckdetails EMERGE.requestdetails DB2INST1.MVRCCRESULT | DB2INST1.iPolicy | EMRGSG018 | BP8(T) BP4(I) |

4.8 CICS Transaction Server v3.2 for z/OS

The LGI scenario application uses CICS and the WebSphere MQ-CICS bridge in a standard way. The programs contain CICS API and EXEC SQL statements that perform queries and updates to a DB2 database. WebSphere Message Broker places a request message on a WebSphere MQ queue, and the WebSphere MQ-CICS bridge invokes the CICS programs.

In this section, we assume that your CICS regions exist. We explain the steps that are necessary to configure and deploy a new application to those existing regions.

4.8.1 Configuration

Our solution takes advantage of standard CICSplex and WebSphere MQ capabilities for availability, reliability, and security. Configuration steps are not unique to this solution.

CICS

If you are service-oriented architecture (SOA)-enabling an existing solution, it is likely that the CICS system resource definitions will be complete. If you are deploying a new application, such as LGI, use the CICS utilities of your choice to define the following types of resources for the application:

- ▶ DB2EDET
- ▶ DB2TDEF
- ▶ FILEDEF
- ▶ LSRDEF
- ▶ PROGDEF

WebSphere MQ-CICS bridge

Use the instructions to install and configure the MQ-CICS bridge as explained in the “Customizing for CICS” topic of the WebSphere MQ 6.0 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqsav.doc/csq83hx.htm>

Customize and use the samples that are provided to define the CICS resources for your environment.

When you define your WebSphere MQ resources, consider the method that WebSphere Message Broker (your enterprise service bus (ESB)) will use to communicate with the MQ-CICS bridge. If WebSphere Message Broker and the

MQ-CICS bridge are both running in z/OS queue managers within the same queue sharing group, define your bridge request queue as a shared queue and run multiple bridge monitors. Setting your queue this way provides the highest level of availability.

However, if the message broker is not running on z/OS or you cannot use shared queues, you must define the channels and related objects that are necessary to communicate between the queue managers. Use WebSphere MQ clustering if possible.

4.8.2 Security configuration

Security for the back-end application components (including CICS) uses a trusted user ID for access to the application programs. RACF is used to secure individual resources as needed. The MQ-CICS bridge is started with LOCAL authentication, meaning that the CICS programs that are run by the bridge task are started with the CICS DFLTUSER ID. There is no checking of other user IDs or passwords.

4.8.3 Application deployment

After the CICS resources and the MQ-CICS bridge are configured, the application can be deployed as follows:

1. Define the VSAM file that is necessary for the quote flow.
2. Create the DB application database.
3. Define the required CICS resources as described previously.
4. SQL precompile, translate, compile, and linkedit each program.
5. Bind the DBRMs that are generated from each program into a DB2 PLAN.

4.9 WebSphere MQ File Transfer Edition on AIX

This section explains how to configure WebSphere MQ File Transfer Edition (FTE) 7.0 to meet the needs of the LGI scenario.

4.9.1 Configuring the AIX coordination queue manager

The coordination queue manager is installed at the back end on a dedicated machine. On the coordination queue manager machines:

1. Connect the coordination queue manager to the agent queue managers. Connect the coordination queue manager to the cluster and define the appropriate SSL artifacts.
2. Define the WebSphere MQ objects that are required by WebSphere MQ FTE.
3. Configure WebSphere MQ security.

Note the following considerations:

- ▶ The coordination queue manager must be running at WebSphere MQ 7.0.
- ▶ A WebSphere MQ FTE installation is not required to configure the coordination queue manager.

4.9.2 Security configuration

In addition to the basic security configuration for WebSphere MQ FTE agents, additional security configuration steps are required to enable communication between agents and the coordination queue manager.

Agents use default channel security (in conjunction with the MCAUSER channel attribute) to put messages into queues on the coordination queue manager. However, some of those messages are then read and published by the publish/subscribe engine in WebSphere MQ. The publish/subscribe engine publishes messages by using the (agent) user specified in the context of the message. This means that the user associated with the agent on the agent machine should be also defined on the coordination queue manager machine and given access to publish on the coordination queue manager.

Tip: WebSphere MQ does not report publish failures that are caused by security failures. When debugging WebSphere MQ FTE problems, messages can be discarded without errors reported. If agents fail to register with the coordination queue manager or file transfers do not function correctly, first check that messages flow across the channels from the agent being administered and the coordination queue manager.

If messages are not flowing, look in the agent logs. If messages are flowing, the coordination queue manager might not be configured to give the agent user access for publish. If the coordination queue manager is not configured this way, the coordination queue manager will not publish the message and will not report an error.

4.9.3 Debugging WebSphere MQ FTE problems

WebSphere MQ FTE agent logs hold status and error messages for agents. The agents also capture first failure data captures (FFDC). For both logs and FFDCs, see the `<WMQ FTE var install root>/<coordination queue manager name>/agents/<agent name>/logs` directory.

WebSphere MQ FTE is a WebSphere MQ application. For details about WebSphere MQ Trace, see “Debugging access control problems” on page 73.

4.9.4 Application deployment

No application deployment is required for WebSphere MQ FTE on AIX.

4.10 WebSphere MQ File Transfer Edition on z/OS

This section explains how to configure WebSphere MQ FTE 7.0 to meet the needs of the LGI scenario.

4.10.1 Configuration of the z/OS agent

The LGI scenario environment contains two back-end queue managers that are mainly used to serve two WebSphere Message Brokers. One of these queue managers also hosts the back-end WebSphere MQ FTE agent.

On the back-end LPAR:

1. Install WebSphere MQ FTE. Choose the client installation option so that binding connections to queue managers are supported.
2. Before creating a WebSphere MQ FTE agent, ensure that the coordination queue manager is available. See the back-end WebSphere MQ FTE section for installing and configuring the coordination queue manager.
3. After the coordination queue manager is configured on the back-end server, create a WebSphere MQ FTE agent.
4. Define the queues that are required by the agent on the local queue manager.
5. Check that the agent is registered with the coordination queue manager.

Note the following considerations:

- ▶ WebSphere MQ FTE runs on UNIX System Services. WebSphere MQ client connections are not supported in a UNIX System Services environment. Therefore, the agent must connect to the z/OS queue manager by using a binding connection.
- ▶ WebSphere MQ FTE agents that connect to z/OS queue managers by using binding connections require WebSphere MQ for z/OS Fix Pack 6.0.2.5.

For step-by-step instructions, see “Instructions for 4.10, ‘WebSphere MQ File Transfer Edition on z/OS’” on page 226.

4.10.2 Security configuration

WebSphere MQ FTE agents benefit from using the existing secured WebSphere MQ infrastructure. For example, the channels between queue managers already exist and are secured with SSL. Therefore, WebSphere MQ FTE messages that are transmitted over those channels are secure.

WebSphere MQ FTE is a WebSphere MQ application. The appropriate steps must be taken to enable access for WebSphere MQ FTE users. Each WebSphere MQ FTE agent runs associated with the user that executed the **fteStartAgent** command. These users require access to WebSphere MQ objects including the following objects:

- ▶ The queue manager object
- ▶ Agent queues (named `SYSTEM.FTE.*.<agent name>`)
- ▶ The `SYSTEM.CLUSTER.TRANSMIT.QUEUE`

Security access is configured by using the RACF on z/OS and by using the OAM on AIX and Linux for System z.

In addition to the basic security configuration described in this section, additional security configuration steps are required to enable communication between agents and the coordination queue manager. For further information, see “Configuring the AIX coordination queue manager” on page 226.

4.10.3 Debugging WebSphere MQ FTE problems

WebSphere MQ FTE agent logs hold status and error messages for agents. The agents also capture FFDC. For both logs and FFDCs, see the `<WMQ FTE var install root>/<coordination queue manager name>/agents/<agent name>/logs` directory (for example, `/var/fte/LGI.BACK.FTE.CQM/agents/BE01/logs`).

WebSphere MQ FTE is a WebSphere MQ application, In the event of access control problems, consult the WebSphere MQ z/OS documentation.

4.10.4 Application deployment

No application deployment is required for WebSphere MQ FTE on z/OS.

4.11 IBM Tivoli Monitoring on AIX

This section explains how to configure IBM Tivoli Monitoring to meet the needs of the LGI scenario.

4.11.1 Configuration of IBM Tivoli Monitoring

The LGI scenario environment contains a dedicated back-end machine to host the IBM Tivoli Monitoring components to support monitoring of the front-end Web applications. The IBM Tivoli Composite Application Manager agents that are installed on the front-end machines will communicate with the IBM Tivoli Monitoring monitoring infrastructure.

On each the back-end machine:

1. Install DB2 (required for data warehousing).
2. Install the components of IBM Tivoli Monitoring (for example, the Tivoli Enterprise Management Server, Tivoli Enterprise Portal Server, and the Tivoli Enterprise Portal).
3. Add the appropriate application support to IBM Tivoli Monitoring components.
4. Check the configuration in the Tivoli Enterprise Portal.

For step-by-step instructions, see “Instructions for 4.11, ‘IBM Tivoli Monitoring on AIX’” on page 228.

4.11.2 Application deployment

No application deployment is required for IBM Tivoli Monitoring on AIX.

Securing integration points

This chapter introduces the security implemented in the Lord General Insurance (LGI) scenario to secure the data that is exchanged between the Web services and their clients. It includes the following topics:

- ▶ “Web Services Security on AIX and z/OS” on page 136
- ▶ “Configuration of the runtime environments” on page 137
- ▶ “Security configuration” on page 137
- ▶ “LGI scenario implementation information” on page 141

5.1 Web Services Security on AIX and z/OS

This section contains an overview of the configuration and deployment tasks that were required to configure Web Services Security (WS-Security) in the LGI scenario. The following information is based on the securing JAX-RPC 1.1 Web services (part of J2EE 1.4) that complies with the 1.0 WS-Security specification. The versions are supported by WebSphere Application Server 6.1 (without the Web services feature pack).

Our implementation does not cover the following information:

- ▶ Encryption of specific parts of a message body
- ▶ Securing Web service calls through intermediaries
- ▶ Issues related to Web services with multiple clients or the dynamic introduction of Web service clients to an existing service
- ▶ Authentication through the use of digital signatures, ID assertion, Lightweight Third Party Authentication (LTPA), or custom tokens

Our implementation is limited to user name tokens.

For more detailed steps about the WS-Security configuration in the LGI scenario, see “Instructions for 2.3, ‘Securing the application’” on page 154.

The LGI scenario implementation

The following Web service calls in the LGI scenario were secured by using WS-Security. Our scenario also demonstrated integration between key WebSphere brand products, which include WebSphere Application Server, WebSphere Process Server, and WebSphere Message Broker:

- ▶ Integrity (signing) and confidentiality (encryption) were configured for the message body of the SOAP/HTTP Web service request. They were also configured for the response between a message flow in WebSphere Message Broker on z/OS (client) and the underwriter Web service hosted on WebSphere Application Server on z/OS (service).
- ▶ Integrity (signing) and confidentiality (encryption) were configured for the message body and user name token of the SOAP/HTTP Web service request. They were also configured for the response between a Business Process Execution Language (BPEL) business flow hosted on WebSphere Process Server on z/OS (client) and the Market Value Reduction (MVR) check Web service hosted on WebSphere Application Server on Windows (service). Basic authentication is achieved by passing a user name token on the request.

5.2 Configuration of the runtime environments

The configuration of the WebSphere Application Server, WebSphere Process Server, and WebSphere Message Broker runtime environments is described in the following sections:

- ▶ 3.4, “WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z” on page 65
- ▶ 4.2, “WebSphere Application Server 6.1 on z/OS” on page 100
- ▶ 4.3, “WebSphere Process Server on z/OS” on page 103
- ▶ 4.6, “WebSphere Message Broker on z/OS” on page 122

5.3 Security configuration

Web Services Security is enabled by defining WS-Security configurations and bindings for both the Web service and its client application. The configuration indicates which parts of the message are to be secured (signed, encrypted, or authenticated). The bindings define how certificates and tokens will be used to secure the message.

The receiving end of Web service call (Web service for requests and client for responses) determines the level of WS-Security that is required. The sending end (client for request and Web service for response) must specify the corresponding definitions to ensure compliance for the message that is sent. The Web service request and response flows are independent and can use different configurations. For example, if sensitive data is sent only in the response, it might not be necessary to encrypt the request.

Support for WS-Security bindings: WebSphere Application Server and WebSphere Process Server support WS-Security bindings that are defined at an application-, server-, or cell-level. This implementation uses application-level bindings. For more information about using WebSphere Application Server or WebSphere Process Server server-level bindings, see “Default Web services security configuration for details” in the WebSphere Application Server Version 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.etools.webservice.security.doc/topics/cwbs_defaultconfigwsssecurity.html

5.3.1 Using public-key cryptography to secure Web services

When a Web service has a single client known at deployment time, public-key cryptography can be used to sign and encrypt the Web service as follows and as shown in Figure 5-1:

- ▶ The Web service client signs requests by using its private key. The Web service verifies the signature by using the client's public key. This allows the Web service to establish that a trusted client sent the request and that it has not been tampered with.
- ▶ The Web service encrypts requests by using the Web service's public key. The Web service decrypts requests by using its own private key, which ensures that the request can only be decrypted by the intended service.
- ▶ The Web service signs responses by using its private key. The signature is verified by the Web service client by using the Web service's public key. This allows the Web service client to establish that a trusted service sent the response and that it has not been tampered with.
- ▶ The Web service encrypts responses by using the Web service client's public key. The Web service client decrypts responses by using its own private key. This ensures that the response can only be decrypted by the intended client.

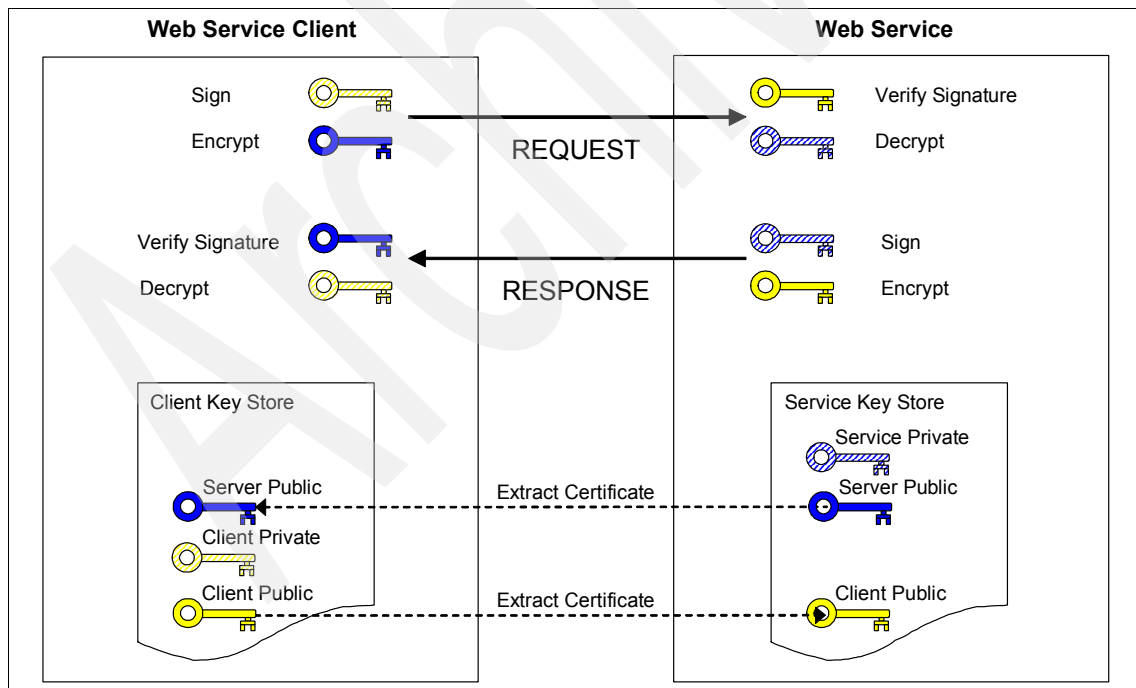


Figure 5-1 Use of keys to sign and encrypt Web service requests and responses

5.3.2 Configuring WS-Security in WebSphere Application Server

The WS-Security configuration and application-level bindings are defined by using an application assembly tool. The following tools are supported:

- ▶ IBM Rational Application Developer 7.0 for J2EE 1.4 Web services and clients
- ▶ WebSphere Integration Developer 6.1 for business process or mediation flow Web service imports and exports

To configure WS-Security by using application-level bindings and public key cryptography in either WebSphere Application Server 6.1 or WebSphere Process Server 6.1:

1. Create (or identify suitable existing) keystores and personal certificates in the application development test environment for use by the Web service and client and exchange their public certificates.
2. Define the WS-Security configuration for the Web service by using an application assembly tool.
3. Define the WS-Security application-level bindings for the Web service by using an application assembly tool.
4. Define the corresponding WS-Security configuration for the Web service client by using an application assembly tool.
5. Define the WS-Security application-level bindings for the Web service client by using an application assembly tool.
6. Deploy and test the secured Web service and client in the application development test environment. Export the enterprise applications that contain their WS-Security configurations and application-level bindings.
7. Create (or identify suitable existing) keystores and personal certificates in the target runtime environments for use by the Web service and client. Exchange their public certificates.
8. Deploy the Web service and Web service client applications to the target runtime environments.
9. Use the Integrated Solutions Console to customize the Web service application-level bindings for the target runtime environment.
10. Use the Integrated Solutions Console to customize the Web service client application-level bindings for the target runtime environment.
11. Test the secured Web service request and response flows.

5.3.3 Configuring WS-Security in WebSphere Message Broker

The WS-Security configuration and application-level bindings are defined at a broker level by using the Policy Set Editor that is available through the Message Broker Toolkit.

To configure WS-Security by using public key cryptography in WebSphere Message Broker 6.1:

1. Create (or identify suitable existing) keystores and personal certificates in the application development test environment for use by the Web service and client. Exchange their public certificates.

Note the following considerations:

- Certificates that are used to sign and encrypt outgoing Web service calls from the broker (requests when the broker is acting as a Web service consumer) must be added to the broker's keystore.
 - Certificates that are used to verify signatures and decrypt incoming Web service calls to the broker (responses when the broker is acting as a Web service consumer) must be added to the broker's truststore.
2. Define the Policy Set (WS-Security configuration) for the broker that is hosting the Web service (provider) by using the Policy Set Editor.
 3. Define the Policy Set Binding (WS-Security binding) for the broker that is hosting the Web service (provider) by using the Policy Set Editor.
 4. Define the corresponding Policy Set (WS-Security configuration) for the broker that is hosting the Web service client (consumer) by using the Policy Set Editor.
 5. Define the Policy Set Binding (WS-Security binding) for the broker that is hosting the Web service client (consumer) by using the Policy Set Editor.
 6. Associate the Policy Set and Policy Set Binding with the Web service (provider) message flow (either at the message flow or node level), and deploy the secured message flow application.
 7. Associate the Policy Set and Policy Set Binding with the Web service client (consumer) message flow (either at the message flow or node level), and deploy the secured message flow application.
 8. Test the secured Web service request and response flows.

5.4 LGI scenario implementation information

This section provides a high level overview of the steps that are required to configure WS-Security for the Web service calls within the LGI scenario.

5.4.1 Configuring WS-Security for the MVR Web service call

To configure WS-Security for the MVR Web service call:

1. Configure authentication (user name token), integrity, and confidentiality for the MVR Check Web service. Define the application-level bindings by using Rational Application Developer.
2. Configure authentication (user name token), integrity, and confidentiality for the BPEL process flow (client to MVR Web service). Define the application-level bindings by using WebSphere Integration Developer.
3. Deploy the MVR Web service to WebSphere Application Server and the BPEL process flow (client) to WebSphere Process Server (with corresponding WS-Security configurations). Customize the WS-Security bindings for the target environments by using the Integrated Solutions Console.
4. Test the MVR Web service by running the LGI scenario by using WebSphere Application Server tracing to confirm that the SOAP/HTTP request and response are signed and encrypted as expected.

5.4.2 Configuring WS-Security for the underwriter Web service call

To configure WS-Security for the underwriter Web service call:

1. Configure integrity and confidentiality for the underwriter Web service, defining application-level bindings by using Rational Application Developer.
2. Configure integrity and confidentiality for the WebSphere Message Broker message flow (client to underwriter Web service) by using the Policy Set and Policy Set Bindings at the broker level.
3. Deploy the underwriter Web service to WebSphere Application Server. Customize the WS-Security bindings for the target WebSphere Application Server environment by using the Integrated Solutions Console and deployed the message flow (client) to WebSphere Message Broker (with corresponding WS-Security configurations).
4. Test the underwriter Web service by running the LGI scenario.

Testing the solution

The Mixed Platform Stack Project was an integration test project. Unlike earlier test phases from unit testing through system verification testing that focus on individual components, integration testing focuses on the entire solution. The integration test integrates new software into a simulated production environment. It targets problems that are related to how different software products and solution components interact.

Service-oriented architecture (SOA) solutions have a high degree of product and component interaction, which makes integration testing particularly important. The project that is described in this document is unique in that the test is performed in two different environments:

- ▶ A z/OS only environment where all application components run on z/OS
- ▶ A mixed platform environment including AIX and z/OS components

The objectives and methodologies are the same for both.

The following sections describe the test methodology and test scenarios that were used to achieve the results that are described in this book. They include the following topics:

- ▶ “Test methodology” on page 144
- ▶ “Test results” on page 148

6.1 Test methodology

To complete our testing, we deployed an existing application into our new test environments. To validate that the software configurations used in the solution can be implemented in the various environments, our test included testing functional and non-functional capabilities of the solution. We completed a function or system regression test.

Our tests validated that the application's business flows worked as designed in the new environments. Security is one of the key non-functional requirements that we validated. We limited security testing to basic authentication and authorization. We performed each of the test scenarios with authorized and non-authorized users to ensure that the application was properly secured.

We did not explicitly test the systems management aspects of the application. We used existing systems management infrastructure to monitor all software products and components. We did not use any new test scenarios.

The goal of our test effort was to validate the interactions of the various products while working together as part of the overall environment. As a result, we paid particular attention to how this particular application and set of software products behaved with other system and application components.

6.1.1 Approach by the z/OS Integration Test Team

This section describes the overall test approach that the z/OS Integration Test Team used for this particular project. While the z/OS Integration Test Team involves a much larger test environment, this particular project emphasized the following elements during testing:

- ▶ Parallel Sysplex exploitation
- ▶ Continuous operation
- ▶ Recovery
- ▶ High utilization and stress
- ▶ Linux processor management by Workload Manager
- ▶ UNIX System Services

The Lord General Insurance (LGI) SOA solution is one of many workloads running in a 24x7 Parallel Sysplex environment. The environment includes coupling facility exploitation over current coupling peer link technologies, Open System Adapter (OSA) networking features, standard channel technologies, exploitation of both general and special purpose processors. At the same time, the environment preserves multiple Parallel Sysplex across rolling initial program

loads (IPLs), CF maintenance, and z/OS service windows with various synchronized timing (9037 or STP) technologies deployed.

z/OS integration testing approach

In addition to the IBM System z hardware and software components, the environment includes an extensive suite of IBM products that make up the test platform. While we did not explicitly test every product during this project, these products are fundamental to the test environment and, therefore, are mentioned here.

The IBM suite includes middleware, security, networking, and system management products and solutions. Some examples are included in the list that follows. However, this is not a complete list and changes often. For a detailed description of the current Parallel Sysplex configuration, see the latest *zSeries Platform Test Report for z/OS and Linux Virtual Servers* document that is available on the z/OS Internet Library at the following address:

<http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/>

Our testing of middleware products in this environment focused on exercising the functionalities that are new to the current releases for the following middleware products:

- ▶ CICS
- ▶ DB2
- ▶ IMS
- ▶ Tivoli Composite Application Manager for WebSphere
- ▶ WebSphere Application Server
- ▶ WebSphere Enterprise Service Bus (ESB)
- ▶ WebSphere Message Broker
- ▶ WebSphere MQ
- ▶ WebSphere MQ File Transfer Edition
- ▶ WebSphere Process Server
- ▶ WebSphere Service Registry and Repository

Our testing of security products and capabilities focused on the following security areas and products:

- ▶ Enterprise Identity Mapping (EIM)
- ▶ Integrated Cryptographic Service Facility (ICSF)
- ▶ IBM HTTP Server (IHS)
- ▶ Lightweight Directory Access Protocol (LDAP)
- ▶ Network Authentication Service (Kerberos)
- ▶ Public Key Infrastructure (PKI) Services
- ▶ Resource Access Control Facility (RACF)
- ▶ System Secure Sockets Layer (SSL)

Our network testing focused on the following products:

- ▶ DFSMS/MVS Network File System (NFS)

The NFS allows workstations to access data that resides on MVS as though it were local to that workstation. It supports file serving of HFS/ZFS and standard MVS data sets.

- ▶ z/OS Communications Server, including TCP/IP and Systems Network Architecture (SNA)

Our objectives for the integration test for middleware, security, and network exploitation included but were not limited to the following items:

- ▶ Validate the documentation that is available for setting up and customizing the current release of z/OS and product environments.
- ▶ Verify that the products work together by using the new dependencies.
- ▶ Run customer-like applications that exploit the enhanced functions.
- ▶ Validate the interaction among the products.
- ▶ Validate the recovery aspects of the products when failures occur.
- ▶ Provide feedback to the appropriate information development groups and the product owners when there is an issue in the documentation or the product.
- ▶ Document experiences with each product in the zSeries Platform Test Report for z/OS and Linux Virtual Servers.

In summary, our focus was on the interactions between the various software products and the z/OS system components.

Mixed platform testing approach

The mixed platform testing followed much the same approach that was used in the z/OS Integration Test approach. The testing ensured that the hardware, operating systems, and software products worked as expected and that the applications deployed to those products worked correctly. A key focus item involved ensuring that the interactions between the products running on platforms included in our scenario performed as expected regardless of the platform specifics.

6.1.2 Test scenarios

The following information provides high-level documentation for the test scenarios that we used to test the environment for this project.

Deployment

The objective of the deployment test scenarios was to verify that the deployment of the LGI solution components could be completed without errors.

The test scenario assumed that the WebSphere Application Server, WebSphere MQ, Message Broker, CICS, and DB2 installation and configuration steps were already completed. The tester performed the steps that were necessary to deploy the entire Quote and Policy process flows. These flows included the WebSphere Application Server front-end components, message flows (ESB), and back-end services in CICS and the WebSphere Application Server. Finally, the tester executed a simple end-to-end test of both the Quote and Policy process flows.

End-to-end test of the quote process flow

The objective of the end-to-end test of the quote process flow was to verify that an end-to-end application flow that includes the invocation of the Quote process can be completed without errors.

Our test scenario assumed that the deployment scenario completed successfully. The tester extensively tested the Quote process and validated the results. Then the tester ran the automated workload at low to normal levels for a typical run of eight hours or more.

End-to-end test of the policy process flow

The objective of the end-to-end test of the policy process flow was to verify that an end-to-end application flow that included invoking the Policy process could be completed successfully.

Our test assumed that the deployment scenario completed successfully. The tester extensively tested the Policy process and validated the results. The tester then ran the automated workload at low to normal levels for a typical run of eight hours or more.

Security tests

The objective of the tests related to security verified that the end-to-end test scenarios described previously could be executed with all security features turned on.

Our test focused on the Execute Quote and Policy end-to-end scenarios with all security features turned on. We conducted each test multiple times with valid users, users that do not exist at all, and existing users that are not authorized to run the processes selected.

6.2 Test results

We completed each of the categories of tests described in 6.1.2, “Test scenarios” on page 146, in both environments. We used the tests to verify that we were able successfully deploy the platforms, products, and applications described in our scenario. We carried out these tests in the same manner as for a system that is being built to go into production. We designed these tests to build confidence that similar systems can be built and deployed in comparable customer environments.

Where we encountered problems, we are able to successfully resolve the problems to continue on to successful completion of the project. In addition, we were able to identify the necessary fixes or additional steps that were required to resolve these problems. These fixes or additional procedures are listed in this report as hints and tips and are integrated in the instructions in this book.

Open issues

This appendix describes the following issues that are currently under investigation by IBM:

- ▶ “ClassCastException customizing WS-Security bindings in WebSphere Process Server 6.1” on page 150
- ▶ “No predefined keystores customizing WS-Security bindings in WebSphere Application Server 6.1” on page 151
- ▶ “Messaging Engines do not start after upgrading WebSphere Application Server to 6.1.0.18” on page 151

ClassCastException customizing WS-Security bindings in WebSphere Process Server 6.1

ClassCastException attempts to configure Web Services Security application-level bindings for business process flow Web service imports or exports by using WebSphere Process Server 6.1 integrated solution console, when the bindings were defined using WebSphere Integration Developer 6.1.

This issue affects WebSphere Integration Developer 6.1 with deployments to WebSphere Process Server 6.1 and WebSphere ESB 6.1.

The workaround is to perform the following tasks:

1. In WebSphere Integration Developer, open the **J2EE perspective**.
2. In the Project Explorer, expand the **JSR-109 Web Services** project and double-click the Web service import (under Clients) or export (under Services). Depending on whether you selected the import or export, the EJB Deployment Descriptor or Web Services Editor opens.
3. From the editor, click the **WS Binding** or **Binding Configurations** tab. In the Service References section, select the appropriate entry and then expand **Security Request Generator Binding Configuration**.
4. Make a dummy change to the bindings:
 - a. Expand **Property**.
 - b. Click **Add** to add a new property and then immediately delete this property.
 - c. Save and close the EJB Deployment Descriptor or Web Services Editor.
5. In the Project Explorer, expand the EJB project for the business process module and then expand the **ejbModule** folder. Expand the **META-INF** folder and open the **ibm-webserviceclient-bnd.xmi** or **ibm-webservices-bnd.xmi** file as appropriate. Check that each XML element has an `xmi:id` tag.
6. Export the business process module as an EAR file as usual.

Note: The new property is not required. Just enough activity in the WS-Security bindings section of either the EJB Deployment Descriptor or Web Services Editor triggers the tooling into thinking a change has been made. This causes the bindings file to be regenerated with the correct `xmi:id` tags. If the business process module code is exported to another workspace as a project interchange file, this workaround must be repeated in the new workspace.

APAR JR31008 has been opened to address this issue. The cause is related to WebSphere Integration Developer's Module Deployment Editor not generating `xmi:id` tags for all elements in the WS-Security binding file (`ibm-webservices-bnd.xmi` or `ibm-webserviceclient-bnd.xmi` in the EJB project generated for the business process module).

No predefined keystores customizing WS-Security bindings in WebSphere Application Server 6.1

When customizing WS-Security bindings by using the WebSphere Application Server 6.1 integrated solutions console, the predefined keystore list is not displayed. This issue affects WebSphere Application Server 6.1 on distributed platforms.

The workaround was to manually input the fully-qualified path to the keystore. However, this issue has been fixed by APAR PK69014 and is targeted for inclusion in WebSphere Application Server 6.1.0.23 and WebSphere Application Server 7.0.0.3. For details, see "PK69014: PREDEFINED KEYSTORE IS NOT LISTED ON ADMINISTRATIVE CONSOLE," at the following address:

http://www-01.ibm.com/support/docview.wss?rs=180&context=SSEQTP&q1=PK69014&uid=swg1PK69014&loc=en_US&cs=utf-8&lang=en

Messaging Engines do not start after upgrading WebSphere Application Server to 6.1.0.18

The WebSphere Application Server adjunct regions issues the message "CWSIS1530E: The data type, 12, was found instead of the expected type, -1, for column, URI, in table, WSRRUSR.SIBCLASSMAP" and the messaging engines do not start.

This issue affects all WebSphere Application Server 6.1.0.18 cells that use an SI Bus, including WebSphere Process Server and WebSphere Service Registry and Repository configurations.

The workaround is to edit the `sib.properties` file for the application server and add the `"sib.msgstore.jdbcPerformColumnChecks=false"` property.

APAR PK74286 has been opened to address this issue.

Detailed instructions

This appendix provides detailed instructions that correspond to the previous sections of this document. It includes the following sections:

- ▶ “Instructions for 2.3, ‘Securing the application’” on page 154
- ▶ “Instructions for 2.4, ‘Securing the development environment’” on page 157
- ▶ “Instructions for 2.5, ‘Application testing’” on page 158
- ▶ “Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’” on page 160
- ▶ “Instructions for 3.4, ‘WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z’” on page 169
- ▶ “Instructions for 3.5, ‘WebSphere MQ on AIX and Linux for System z’” on page 183
- ▶ “Instructions for 3.6, ‘DB2 Enterprise 9 on AIX and Linux for System z’” on page 192
- ▶ “Instructions for 3.7, ‘IBM HTTP Server 6.1 on AIX’” on page 194
- ▶ “Instructions for 3.9, ‘WebSphere MQ on z/OS’” on page 195
- ▶ “Instructions for 3.11, ‘WebSphere MQ File Transfer Edition on AIX’” on page 199
- ▶ “Instructions for 3.12, ‘IBM Tivoli Composite Application Manager on AIX’” on page 201

- ▶ “Instructions for 4.3, ‘WebSphere Process Server on z/OS’” on page 204
- ▶ “Instructions for 4.4, ‘WebSphere Service Registry and Repository on z/OS’” on page 210
- ▶ “Instructions for 4.5, ‘WebSphere MQ on z/OS’” on page 213
- ▶ “Instructions for 4.6, ‘WebSphere Message Broker on z/OS’” on page 222
- ▶ “Instructions for 4.9, ‘WebSphere MQ File Transfer Edition on AIX’” on page 225
- ▶ “Instructions for 4.10, ‘WebSphere MQ File Transfer Edition on z/OS’” on page 226
- ▶ “Instructions for 4.11, ‘IBM Tivoli Monitoring on AIX’” on page 228
- ▶ “Instructions for 5.1, ‘Web Services Security on AIX and z/OS’” on page 236

Instructions for 2.3, ‘Securing the application’

This section contains detailed instructions for the steps to secure the application. For a high level view of this information, see 2.3, “Securing the application” on page 42.

To configure the Web application so that both the secure and non-secure URLs invoke the same servlet (for example, `LGISecureServlet`):

1. Open the servlet configuration properties of the Web module:
 - a. Open the **J2EE perspective** of the IBM Rational Application Developer workspace.
 - b. In the Project Explorer, open the deployment descriptor for the Web project.
 - c. Click the **Servlets** tab.
2. Add a URL mapping for a secure URL:
 - a. In Servlets and JSP, click **LGISecureServlet**.
 - b. In URL Mappings, ensure that a default mapping already exists, such as **/LGIServlet → LGISecureServlet**.
 - c. In URL Mappings, select **Add**.
 - d. For Servlet name, select **LGISecureServlet**.
 - e. For URL Pattern, type `/secure/LGIServlet`.
 - f. Click **Finish**.
 - g. Close the deployment descriptor.

To configure the login information for the Web application:

1. Open the login configuration properties of the Web module:
 - a. Open the **J2EE perspective** of the Rational Application Developer workspace.
 - b. In the Project Explorer, open the deployment descriptor for the Web project.
 - c. Click the **Pages** tab.
2. In the Login pane, configure the Web application to use form-based login and set the JSP values:
 - a. For Authentication method, select **FORM**.
 - b. For Login page, type a value (for example, /login.jsp). JSP must exist.
 - c. For Error page, enter a value (for example, /login-error.jsp). JSP must exist.

Enabling role-based security in a Web application

The following task defines two security roles to a dynamic Web project for use in the Lord General Insurance (LGI) application. This task also restricts the use of Web pages associated with the secure URL to one of these roles.

1. Open the security configuration properties of the Web module:
 - a. Open the **J2EE perspective** of the Rational Application Developer workspace.
 - b. In the Project Explorer, open the deployment descriptor for the Web project.
 - c. Click the **Security** tab.
2. Define the roles used by the LGI application Web module:
 - a. In Security Roles, click **Add**.
 - b. For Name, type All Quote Users, and for Description, type Any user with a registered account.
 - c. Click **Finish**.
3. Repeat step 2. However, for Name, type Staff Discount, and for Description, type This provides a discount to members of staff applying for insurance.
4. Restrict the HTTP methods for the secure URL:
 - a. In Security Constraints, select **Add**.
 - b. For Constraint name, type Secure Pages Constraint and click **Next**.

- c. For Resource name, type Secure Resources. Optionally, enter a description. Then under HTTP Methods, select all values that are displayed.
 - d. For Pattern, select **Add**, for Name, type /secure/*.
 - e. Click **OK**.
 - f. Click **Finish**.
5. Define the roles that can access the restricted HTTP methods:
 - a. For Authorized Roles, select **Add**.
 - b. For Description, type All Quote Users. Under Role Name, select the **All Quote Users** role from the list.
 - c. Click **Finish**.
 6. Configure the client server connection to secure the data content while it is in transit by using Secure Sockets Layer (SSL). (The content includes user IDs and passwords). To do this, go to User Data Constraint, and for Type, select **CONFIDENTIAL**.

To add a security role to an EJB project for use in the LGI application and restricts selected bean methods to that role:

1. Open the assembly configuration properties of the EJB module:
 - a. Open the **J2EE perspective** of the Rational Application Developer workspace.
 - b. In the Project Explorer, open the deployment descriptor for the EJB project.
 - c. Click the **Assembly** tab.
 - d. Define the security role:
 - i. For Security Roles, click **Add**.
 - ii. For Name, type EJB Quote User.
 - iii. Optional: For Description, enter a value.
 - e. Click **Finish**.
2. Restrict the selected methods to the security role.
 - a. For Method Permissions, select **Add**.
 - b. Select the **Security Roles** radio button, and from the list of Security roles found, select **EJB Quote User**. Click **Next**.
 - c. On the Enterprise Bean Selection page, select the beans that are to be restricted to the EJB Quote User role, for example: **RetrieveStoredQuoteSecureEJB**, **StoreQuoteSecureEJB**, and **UserInfoSecureEJB**. Then click **Next**.

- d. On the Method Elements page, select the methods of the beans selected in previous step that are to be restricted to the EJB Quote User role. For example, select **choose all enterprise bean methods** and click **Finish**.
- e. Close the deployment descriptor.

Instructions for 2.4, ‘Securing the development environment’

This section contains detailed instructions to secure the development environment. For a high level view of this information, see 2.4, “Securing the development environment” on page 45.

Enabling administration security using the local operating system

The following task enables administration security in the WebSphere Application Server 6.1 test environment on Rational Application Developer. The local operating system is used for the user account repository.

1. Open the administrative console for the application server:
 - a. Open the **Servers** view in the Rational Application Developer workspace.
 - b. Select the server (called **IBM WebSphere Application Server v6.1** by default) and start it.
 - c. Right-click the server name (for example, **IBM WebSphere Application Server v6.1**) and select **Run administrative console**.
 - d. Optional: From the administrative console, enter a user ID to identify the user of this administration session. Because administration security is not yet enabled on this server, no authentication or authorization is performed against the value entered at this point.
2. Configure the server to use local operating security:
 - a. Select **Security**.
 - b. Select **Secure administration, application, and infrastructure**.
 - c. For Available realm definitions, select **Local operating system** and click **Configure**.
 - d. For Primary administrative user name, type `wasadmin`, where `wasadmin` is defined as a user account on a local Windows machine.

- e. For Server user identity, click the **Server identity that is stored in the repository** radio button. Enter wasadmin and its password in the appropriate input fields.
 - f. Click **OK**.
 3. To configure Administration and Application Security, from the Select Secure administration, application and infrastructure page:
 - a. Select **Enable administrative security**.
 - b. Ensure that **Enable application security** is also selected. This happens by default when administrative security is enabled.
 - c. Clear **Use Java 2 security to restrict application access to local resources**. Again, this happens by default when administrative security is enabled but the LGI application does not require it.
 - d. For Administrative security, select **Administrative User Roles** and click **Add**.
 - e. For User, type wasadmin and select **All available Role(s)**.
 - f. Click **OK**.
 - g. Select **Apply** and **Save**.
 4. Restart the application server for changes to take effect.
 5. Open the administrative console, enter wasadmin and its password for User ID and password. Confirm that the login is successful.

Instructions for 2.5, 'Application testing'

This section contains detailed instructions for application testing. For a high level view of this information, see 2.5, "Application testing" on page 47.

Testing the session and entity beans

To test the session and entity beans:

1. Create a local Windows version of the DB2 database that the LGI Web application accesses. Load any required tables with test data.
2. Start the WebSphere Application Server 6.1 server.

3. Log on to the administrative console:
 - a. From the Servers view in the Rational Application Developer workbench, right-click the server name and select **Run administrative console**.
 - b. From the logon panel, if administration security is enabled for the server, log in using an authorized user ID. Otherwise, enter an arbitrary value for User ID to uniquely identify the user at this console.
4. From the administrative console, define a Java Database Connectivity (JDBC) connection to the DB2 database. Use the Test connection function to ensure that the application server can access the database.
5. Deploy the Web application to the WebSphere Application Server 6.1 server:
 - a. From the Servers view in the Rational Application Developer workbench, right-click the server name and select **Add and Remove Projects**.
 - b. From the list of Available projects, select the application (for example, **LGDSecureQuoteEAR**) and click **Add** to add it to list of Configured projects.
 - c. Click **Finish**.
6. Use the Universal Test Client to invoke the entity and session beans directly (independently of the servlets and JSP) by following the instructions in the “Testing enterprise beans in the universal test client” section in the WebSphere Application Server 6.1 Information Center at the following address:
<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.ws.ast.st.common.ui.doc/topics/twejbtc.html>

Testing the security roles

To test the security roles:

1. Start the WebSphere Application Server 6.1 server.
2. Log on to the administrative console:
 - a. From the Servers view in the Rational Application Developer workbench, right-click the server name and select **Run administrative console**.
 - b. Log on. If administration security is enabled, log on using an authorized user ID.
3. Enable WebSphere administration and application security by using the local operating system. Use the instructions in the previous section of this document if administration and application security are not already enabled.

4. Deploy the Web application to the application server:
 - a. From the Servers view in the Rational Application Developer workbench, right-click the server name and select **Add and Remove Projects**.
 - b. From the list of Available projects, select the application (for example, **LGDSecureQuoteEAR**) and click **Add** to add it to list of Configured projects.
 - c. Click **Finish**.
5. Open the Web application URL by using your preferred browser. You can also click the Web browser icon on the J2EE perspective or the Web perspective toolbar.
6. Run the Web application by using various user IDs and ensure that the application performs as expected. Consider the following examples:
 - Log on to the application as an unauthorized user. The attempt to log on is rejected.
 - Log on to the application as an authorized user with restricted access. The user can access functions where authorized, but is rejected when attempting to access unauthorized functions.
 - Log on to the application as a fully authorized user. The user can access all authorized functions.

Instructions for 3.2, ‘Tivoli security on AIX and Linux for System z’

This section contains detailed instructions to configure Tivoli Security. For a high level view of this information, see 3.2, “Tivoli security on AIX and Linux for System z” on page 52.

Installation

This section provides detailed information for installing Tivoli Directory Server on AIX or Linux for System z.

Preparing the operating system

To prepare the operating system for installation:

1. Turn on asynchronous I/O, which is required for a directory server:
 - a. Type the following command:
`smitty chgaio`
 - b. For STATE to be configured at system restart, change the setting from *defined* to **available**.
 - c. Type the following command:
`smitty aio`
 - d. Move the cursor to **Configure defined Asynchronous I/O**.
 - e. Click **Enter**.
2. Create a group and a user ID for the directory server instance by entering the following commands in the order shown:
`mkgroup idsldap`
`mkuser pgrp=idsldap home=/home/idsldap shell=/bin/ksh idsldap`
3. Set a password for the new user. Substitute `pwd` with a password appropriate for your environment:
 - a. Type the following command:
`passwd idsldap`
 - b. Enter the `idsldap`'s new password of `pwd`.
 - c. Enter the new password again of `pwd`.
4. Add `root` as a member of the new group
`/usr/bin/chgrpmem -m + root idsldap`
5. Ensure that there is enough free space in the file systems:
 - a. To see the current space in the file systems in kilobytes, type **df -k**.
 - b. Expand the file systems to see the required free space, which is shown as follows:
 - `/var` 100 MB
 - `/tmp` 450 MB (in addition to space for downloading the product code)
 - `/usr` 200 MB
 - `/home` 80 MB for the database

Installing the directory server

To install the directory server by using the InstallShield graphical user interface (GUI):

1. Download the product code into the `/tmp/TDS61` directory by obtaining the following electronic software download part numbers:
 - C12UVML - Tivoli Directory Client-Server with entitlement
 - C12UYML - WebSphere and Tivoli Directory Integrator
 - C12UXML - Optional DB2, not required since it is already installed
 - C12UZML - Optional white pages
2. Expand each downloaded tar file as follows by using the `tar` command from an AIX console:

```
tar -xvf C12UVML.tar
tar -xvf C12UYML.tar
tar -xvf C12UXML.tar
tar -xvf C12UZML.tar
```
3. Remove the tar files after expanding them to make space available in the file system.
4. Download the current recommended version of the Global Security Kit (GSK) and replace the version in the expanded directories of Tivoli Directory Server code:
 - a. Download the `gksa.rte` (64 bit) file from GSKit 7.0.4.11 into the `/tmp/TDS61/tdsV6.1/gskit` directory.
 - b. Download the `gskta.rte` (32-bit) file from GSKit 7.0.4.11 into the `/tmp/TDS61/tdsV6.1/gskit/gsk7bas32` directory.
5. Install Tivoli Directory Server 6.1:
 - a. Type the following command:

```
cd /tmp/TDS61/tdsV6.1/tds
```
 - b. Launch the IBM Tivoli Directory Server V6.1 installation program:

```
./install_tds.sh
```
 - c. For the language, select **English** for this wizard, and click **OK**.
 - d. On the International Program License Agreement page, select the **I accept both the IBM and the non-IBM terms** radio button, and click **Next**.
6. Review the list of applications that have already been installed (in this case, DB2 9.1.0.4 /opt/IBM/db2/V9.1). Click **Next**.
7. For the installation type, select **Custom**, and click **Next**.

8. When you see the features for IBM Tivoli Directory Server 6.1 installation are displayed for selection, leave everything selected except DB2. DB2 is already installed. Then click **Next**.
9. Review the settings and select **Install**.
10. Watch for the “Installation is now complete” message. In addition, ensure that the configuration tool opens. Select **Finish Installation**.

Configuring the directory server instance

Create a new directory server instance by using the IBM Tivoli Directory Server Instance Administration Tool that opened at the end of the previous task:

1. Select **Create**.
2. Select **Create default instance** and click **Next**.
3. Enter the following details:
 - a. For user password, type *pwd*.
 - b. Confirm the password by typing *pwd*.
 - c. For Encryption seed, type *seed*. Be sure to save this in a safe place.
 - d. Confirm the encryption seed: *seed*.
 - e. For Administrator DN password, type *pwd*.
 - f. Confirm password by typing *pwd*.
 - g. Click **Next**.
4. On the Verify Settings page, click **Finish** to begin the instance creation. Messages are displayed showing progress as the instance is created. This includes information about ports and the database creation. The final message is “Task completed.”

5. Review the Results page (Figure B-1). Then click **Close**. The Administration Tool now shows the created instance.

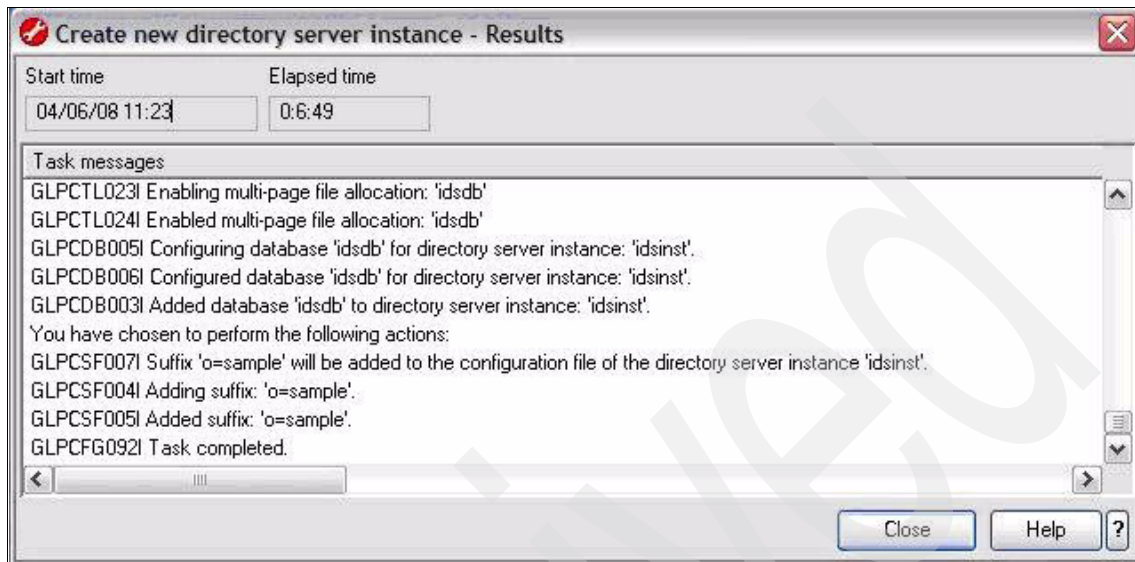


Figure B-1 Results page

6. Click **View** to see details of the instance.
7. Click **Configure** to manage the instance.
8. Select **File** → **Close** to close the Configuration Tool.
9. Click **Close** to close the Administration Tool.
10. To see the instance creation and administration log, type the following command:

```
cd /var/idsldap/V6.1 and view idsadm.log
```

Starting the directory server instance

To start the directory server instance:

1. Change the directory:

```
cd /opt/IBM/ldap/V6.1/sbin/
```
2. Type the following command:

```
./idsslapd -I idsinst
```

Messages are displayed on the panel and written to a log.

3. Enter the following commands:

```
cd /home/idsinst/idsslapd-idsinst/logs
view ibmslapd.log.
```

4. To see the instance configuration, enter the following commands:

```
cd /home/idsinst/idsslapd-idsinst/etc
view ibmslapd.conf
```

5. To check if the instance is running:

```
ps -ef | grep ibmslapd
```

Starting the administration daemon

Start the administration daemon to allow remote management of the directory server instance:

1. Enter the following commands:

```
cd /opt/IBM/lldap/V6.1/sbin
./idsdiradm -I idsinst
```

The administration daemon is started when the directory server instance is created. In addition, an entry is created in `inittab` that starts the daemon when the AIX box is rebooted.

2. To see if the daemon is running:

```
ps -ef | grep ibmdiradm
```

Starting the Web application server

Start the Web application server so that you can use the Web Administration Tool:

1. Enter the following commands:

```
cd /opt/IBM/lldap/V6.1/appsrv/profiles/TDSWebAdminProfile/bin
./startServer.sh server1
```

The server is started when the directory server is installed by using the GUI.

2. To check for the message “Server server1 open for e-business”:

```
cd
/opt/IBM/lldap/V6.1/appsrv/profiles/TDSWebAdminProfile/logs/server1
view SystemOut.log
```

3. To determine if server1 is running:

```
ps -ef | grep server1
```

Starting the Web Administration Tool

After the Web application server is started, start the Web Administration Tool:

1. In a Web browser, type the following URL, substituting the host name of the directory server:

`http://hostname:12100/IDSWebApp/IDSjsp/Login.jsp`

2. In The Tivoli Directory Server Web Administration Tool, select **Console administration login**.
3. For User ID, type superadmin.
4. For Password, type secret.
5. Select **Login** to view the Console Administration Introduction.

Adding the directory server to the Web Administration Tool

Add the directory server to the Web Administration Tool so that it can be administered:

1. In the navigation panel on the left, expand **Console Administration**.
2. Select **Manage console servers**.
3. Click **Add**.
4. Complete the following steps:
 - a. In the Hostname area, enter the host name of the directory server.
 - b. Type Port: 389 (the default non secure port).
 - c. Type Administration port: 3538 (the default non-secure port).
 - d. Click **OK**.
 - e. View the following message:

The directory server *hostname* that listens on 389 was successfully added to the Web Administration Tool. The directory server is now available from the Web Administration Tool login panel.
 - f. Click **OK**. The directory server is now displayed in the list of servers.
5. In the left pane, select **Change console administrator password** to change the password from the initial password (that is, secret):
 - a. For the current password, type secret.
 - b. For the new password, type *pwd*.
 - c. Enter the new password again to confirm it by typing *pwd*.
 - d. Click **OK**.
6. In the navigation panel on the left, click **Logout**. The “Logout successful” message is displayed.
7. Click the **here** link to display the Directory server login page.

8. For LDAP Hostname, type *hostname*:389.
9. For User DN, type *cn=root*.
10. For Password, type *pwd*.
11. Click **Login**. The Directory Server Introduction is displayed.
12. Expand entries in the navigation pane on the left to see usage of the Web administration tool.
13. Click **Logout** to exit the Web Administration Tool.

Building the LDAP structure

To build the Lightweight Directory Access Protocol (LDAP) structure:

1. Log in to the directory server that is added to the Web Administration Tool as described previously.
2. Create the suffix. A suffix (also known as a *naming context*) is a distinguished name (DN) that identifies the top entry in a locally held directory hierarchy. Because of the relative naming scheme used in LDAP, this DN is also the suffix of every other entry within that directory hierarchy. This definition is in the “Configuration” topic within the “Managing suffixes” section of the Tivoli Directory Server Version 6.1 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDS.doc/install16.htm#mgsuff>

To create the suffix:

- a. In the navigation pane on the left, expand **Server administration**.
 - b. Select **Manage server properties**.
 - c. Select **Suffixes**.
 - d. For Suffix DN, type *dc=LGI, dc=com*.
 - e. Click **Add** to add the suffix to the list.
 - f. Click **OK**.
3. Create the domain:
 - a. In the left navigation pane, expand **Directory management**.
 - b. Select **Manage entries**.
 - c. Click **Add** to define the domain.
 - d. From the Structural object classes list, select **domain**. Click **Next**.
 - e. View the Select auxiliary object classes page. Click **Next**.
 - f. On the Required attributes page:
 - i. Type Relative DN: *dc=lgi,dc=com*.
 - ii. Type *dc: LGI*.
 - iii. Click **Next**.

- g. Review the Optional attributes page and click **Finish**.

The top level of the LGI structure in LDAP is displayed as `dc=lgi,dc=com`, which has Object class set to *domain*.

4. Create the groups:
 - a. Click **Add**.
 - b. From the list of Structural object classes, select **organizationalUnit**. Click **Next**.
 - c. View the Select auxiliary object classes page and click **Next**.
 - d. On the Required attributes page:
 - i. Type Relative DN: `ou=groups`.
 - ii. Click **Browse** to the right of Parent DN.
 - iii. Select **dc=lgi,dc=com** and click **Select**.
 - iv. Type `ou: groups`.
 - v. Click **Next**.
 - e. Review the Optional attributes page and click **Finish**.
5. Create the users:
 - a. Click **Add**.
 - b. From the list of Structural object classes, select **organizationalUnit**. Click **Next**.
 - c. View the Select auxiliary object classes page and click **Next**.
 - d. On the Required attributes page:
 - i. Type Relative DN: `ou=users`.
 - ii. Click **Browse** to the right of Parent DN.
 - iii. Select **dc=lgi,dc=com** and click **Select**.
 - iv. Type `ou: users`.
 - v. Click **Next**.
 - e. Review the Optional attributes page and click **Finish**.
6. Continue to build the rest of the LDAP structure for LGI under groups and users.

Instructions for 3.4, 'WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z'

This section contains detailed instructions for configuring WebSphere Application Server 6.1 Network Deployment on AIX. For a high level view of this information, see 3.4, "WebSphere Application Server V6.1 Network Deployment on AIX and Linux for System z" on page 65.

Configuring security

This section provides the steps that we followed to complete our security configuration.

Configuring a user account repository

In this task, note the following details:

- ▶ The primary administrative user is `wasadmin`.
- ▶ The host name of the deployment manager is `dmgrHostname`.
- ▶ The HTTPS port number for the administrative console is `dmgrPort`.
- ▶ The host name of the LDAP server is `ldapHostname`.
- ▶ The port used by the LDAP server is `ldapPort`. This is normally 389.
- ▶ The base distinguished name is `dc=lg,dc=com`. This is used to limit LDAP searches only to the `dc=lg,dc=com` tree.
- ▶ The bind distinguished name is `uid=wasadmin,ou=serviceids,ou=users,dc=lg,dc=com`. This is the distinguished name of the user ID that will be used to bind to the LDAP server.
- ▶ The user filter is configured to search for users by using the `uid` of LDAP objects with an object class of `inetOrgPerson`. This means that the user name with which a users logs in is the `uid` that is defined for that user in LDAP.
- ▶ The group filter is configured to search for groups with LDAP object classes of `groupOfNames`, `groupOfUniqueNames`, or `groupOfURLs`. This means that WebSphere Application Server will look for any group that is of one of these object classes.

To configure WebSphere Application Server to use LDAP as its user account repository:

1. Access the administrative console on the deployment manager.
2. If administrative security is already enabled, log in with a valid user ID and password.

3. Select **Security**.
4. Select **Secure administration, applications, and infrastructure**.
5. From Available realm definitions, select **Standalone LDAP registry** and click **Configure**.
6. On the General Properties page:
 - a. For the Primary administrative user name, type `wasadmin`.
 - b. Select **Automatically generated server identity**. In this case, a cell consists only of WebSphere 6.1 servers. If older servers are present in the cell, you must specify a server identity that is stored in the repository.
 - c. From Type of LDAP server, select **IBM Tivoli Directory Server**.
 - d. For Host, type `ldapHostname`.
 - e. For Port, type `ldapPort`.
7. Select **Advanced Lightweight Directory Access Protocol (LDAP) user registry settings**.
8. On the General Properties page:
 - a. For User filter, type `(&(uid=%v)(objectclass=inetOrgPerson))`.
 - b. For Group filter, type:
`(&(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfUniqueNames)(objectclass=groupOfURLs)))`.
 - c. For User ID map, select the default `(*:uid)`.
 - d. For Group ID map, select the default `(*:cn)`.
 - e. For Group member ID map, accept the default, which is **ibm-allGroups:member;ibm-allGroups:uniqueMember**.
 - f. Click OK to save the changes.
9. Click **OK**. The new settings are validated, and any problems are displayed in the Messages area. If any problems are present, correct them.
10. After the settings are validated and you return to the panel in step 4, ensure that **Standalone LDAP registry** is selected from Available realm definitions. Select **Set as current**. Click **Apply**.
11. Click **Save directly to the master configuration**.
12. Click **Save**.

Enabling administrative security

To enable administrative security:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **Security administration, applications and infrastructure**.
4. Ensure that your desired user account repository is shown in the Current realm definition box.
5. Select **Enable administrative security**.
6. Click **Apply**.
7. Click **Save directly to the master configuration**.
8. Click **Save**.

Administrative security is now enabled. It is now possible to grant different levels of administrative access to users and groups. To grant a user an administrative role:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **Security administration, applications and infrastructure**.
4. Select **Administrative User Roles**.
5. Click **Add**.
6. Add roles to a user:
 - a. In the User field, enter a user ID. The user must exist in the currently active user account repository.
 - b. From Role(s), select the desired role for the user. To select multiple roles, hold down the Ctrl key while selecting each desired role.
 - c. Click **OK**.
7. Click **Save directly to the master configuration**.
8. Click **Save**.

To grant a group an administrative role:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **Security administration, applications and infrastructure**.
4. Select **Administrative Group Roles**.
5. Click **Add**.

6. Add roles to a group:
 - a. Enter a group name. The group must exist in the currently active user account repository.
 - b. From Role(s), select the desired role for the group. To select multiple roles, hold down the Ctrl key while selecting on each desired role.
 - c. Click **OK**.
7. Click **Save directly to the master configuration**.
8. Click **Save**.

Enabling application security

To enable application security:

1. Log in to the administrative console on the deployment manager.
2. Select **Security**.
3. Select **Secure administration, applications, and infrastructure**.
4. Verify that **Administrative security** is enabled.
5. Select **Application security**.
6. Click **Apply**.
7. Click **Save directly to the master configuration**.
8. Click **Save**.

Enabling manual node synchronization

To synchronize a node manually:

1. Ensure that deployment manager is running.
2. Log in to the operating system of the node that you want to synchronize.
3. Change to the profiles directory of the node that you want to synchronize. For example, type the following command:

```
cd /usr/IBM/WebSphere/AppServer/profiles/default/bin
```

4. Type the following command:

```
syncNode.sh dmgrHostname dmgrPort
```

Changing the WebSphere Application Server process identity

Note the following details for this task:

- ▶ The operating system user that WebSphere Application Server will run as a non-root user is `wasuser`.
- ▶ `Wasuser` is a member of the operating system group `wasgroup`.
- ▶ The profile that you want to run as a non-root user is `profilename`.

- The installation directory for WebSphere Application Server is
/usr/IBM/WebSphere/AppServer.

Change the ownership of a profile directory to a non-root user:

1. Log in to the operating system as root.
2. Change the user and group ownership of the profile to wasuser and wasgroup by entering the following command:

```
chown -R wasuser:wasgroup  
/usr/IBM/WebSphere/AppServer/profiles/profilename
```

In an environment with multiple servers, set all WebSphere Application Server processes to run as the same user.

You can use the following steps to ensure that the processes will always run as the desired non-root user even when started by root. Note the following details for this task:

- The operating system user that WebSphere Application Server will run as a non-root user is wasuser.
- Wasuser is a member of the operating system group wasgroup.

To change these settings for a deployment manager:

1. Access the administrative console on the deployment manager.
2. Select **System administration**.
3. Select **Deployment manager**.
4. Select **Java and Process Management**.
5. Select **Process Definition**.
6. Select **Process Execution**.
7. Complete the General Properties page:
 - a. For Run As User, enter wasuser.
 - b. For Run As Group, enter wasgroup.
 - c. Click **OK**.
8. Click **Save directly to the master configuration**.
9. Click **Save**.
10. Restart the deployment manager.

The process to change these settings for node agents, deployment managers, and application servers is the same. However, the path to the Process Execution settings differs for each process type.

To locate the Process Execution settings:

- ▶ For deployment managers, click **System Administration** → **Deployment manager** → **Java and Process Management** → **Process Definition** → **Process Execution**.
- ▶ For node agents, click **System Administration** → **Node agents** → **nodeagent** → **Java and Process Management** → **Process Definition** → **Process Execution**.
- ▶ For application servers, click **Servers** → **Application Servers** → **server** → **Process Execution**.

Whenever you change the process execution settings for a particular component, restart that component to ensure that the new settings are active.

Using SSL between LDAP and WebSphere Application Server

In the steps in this section, we assume that SSL is already enabled on the LDAP server and that you have exported the LDAP public key as a base 64-encoded ASCII file called `LDAPPublicKey.arm`.

The `LDAPPublicKey.arm` file should be copied to the deployment manager server and placed in the `/usr/IBM/WebSphere/AppServer/profiles/dmgr01/config` directory. This file is referred to inside the administrative console as `${CONFIG_ROOT}/LDAPPublicKey.arm`.

Communication over SSL uses SSL configurations (also known as *repertoires*) that define which key and truststores to use for communication. It is possible to use the default SSL configurations, keystores, and truststores that were created when WebSphere Application Server was installed. In these examples, a new SSL configuration, keystore, and truststore are created to make it clear what is required to enable SSL between the servers.

The process used to enable SSL between WebSphere Application Server and the LDAP server includes the following phases:

1. Create a keystore.
2. Create a truststore.
3. Import the public certificate for the LDAP server.
4. Create an SSL configuration.
5. Configure WebSphere Application Server to use the SSL configuration to communicate with the LDAP.

6. Optional: Create a self-signed certificate to identify WebSphere Application Server to the LDAP server. This is only necessary when client authentication is enabled on the LDAP server.

These phases are described in detail in the sections that follow.

Creating a keystore

To create a keystore:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **Key stores and certificates**.
5. Click **New**.
6. On the General Properties page:
 - a. For Name, type LDAPKeyStore.
 - b. For Path, type `${CONFIG_ROOT}/cells/cellname/LDAPKeyStore.p12` to place the keystore in the configuration directory of the deployment manager at a cell level. This path will be synchronized with all nodes in the cell, enabling all nodes to access the keystore.
 - c. For Password and Confirm Password, type a suitable password. This password is used to open the keystore.
 - d. For Type, select **PKCS12**.
 - e. Click **OK**.

Creating a truststore

To create a truststore:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **Key stores and certificates**.
5. Click **New**.
6. On the General Properties page:
 - a. For Name, type LDAPTrustStore.
 - b. For Path, type `${CONFIG_ROOT}/cells/cellname/ LDAPTrustStore.p12` to place the keystore in the configuration directory of the deployment manager at a cell level. This path is synchronized with all nodes in the cell, enabling all nodes to access the keystore.
 - c. For Password and Confirm Password, type a suitable password. This password is used to open the keystore.

- d. For Type, select **PKCS12**.
- e. Click **OK**.

Importing the public key for the LDAP server to the truststore

After the keystore and truststores are created, import the public key for the LDAP server to the truststore in either of the following ways:

- ▶ Import a certificate from a file
- ▶ Retrieve a certificate from a port

These methods are described in the sections that follow.

Importing a certificate from a file

To import a certificate from a file:

1. In the administrative console, navigate to **Key stores and certificates**.
2. Click **LDAPTrustStore**, which is the truststore created earlier.
3. Select **Signer certificates**.
4. Click **Add**.
5. On the General Properties page:
 - a. For Alias, type LDAP Public Key.
 - b. For Filename, type `${CONFIG_ROOT}/LDAPPublicKey.arm`.
 - c. For Data type, select **Base64-encoded ASCII data**.
 - d. Click **OK**.
6. Click **Save directly to the master configuration**.

Retrieving a certificate from a port

To retrieve a certificate from a port:

1. Follow steps 1 through 3 for importing a certificate from a file.
2. Select **Retrieve from port**.
3. For Host, enter the host name of the LDAP server.
4. For Port, type the ldaps port, which is the SSL-enabled port on the LDAP server and is usually port 636.
5. For Alias, type LDAP Public Key.
6. Select **Retrieve signer information**.
7. Verify that the retrieved signer information is correct.
8. Click **OK**.
9. Click **Save directly to the master configuration**.

Creating an SSL configuration

After the keystore and truststore are created and the appropriate certificate is added to the truststore, create an SSL configuration to use them. The SSL configuration is used to determine which truststore and keystore to use for creating the SSL connection. A separate SSL configuration can be created and used specifically for communication with the LDAP server over SSL. All application servers in the cell communicate with the LDAP server. Therefore, a cell scoped SSL configuration is used.

To create an SSL configuration:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **SSL configurations**.
5. Click **New** to create a new cell scoped SSL configuration.
6. On the General Properties page:
 - a. For Name, type `LDAPSSLSettings`.
 - b. For Trust store name, select **LDAPTrustStore**.
 - c. For Key store Name, select **LDAPKeyStore**.
 - d. Click **Get Certificate aliases**.
 - e. Click **OK**.
 - f. Click **Save directly to the master configuration**.

Configuring WebSphere Application Server for LDAP

Finally, configure WebSphere Application Server to use the new SSL configuration for communication with the LDAP server:

1. Access the administrative console.
2. Select **Security**.
3. Select **Secure administration, applications and infrastructure**.
4. For Available realm definitions, ensure that **Standalone LDAP registry** is selected.
5. Select **Configure**.
6. Select **SSL enabled**.
7. Select **Use specific SSL alias**, and then select **LDAPSSLSettings** from the list.
8. Click **OK**.
9. Select **Save to master repository**.
10. Click **Save**.

11. Restart deployment manager, all servers, and node agents to pick up the new settings.

The following task is only necessary if client authentication has been enabled on the LDAP server. In this task, a self-signed certificate is created that is used to identify the application servers to the LDAP server.

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **SSL configurations**.
5. Select **LDAPSSLSettings**.
6. Select **Key stores and certificates**.
7. Select **LDAPKeyStore**.
8. Select **Personal certificates**.
9. Select **Create a self-signed certificate**.
10. On the General Properties page:
 - a. For Alias, type WAS LDAP self signed certificate.
 - b. From the Key size list, select **1024**.
 - c. For Common name, type the host name of your server.
 - d. For Validity period, type an appropriate validity period, which is the number of days for which the certificate will be valid. Secure communication is not possible after the certificate expires, and a new certificate must be generated.
 - e. Click **OK**.
11. Extract the self-signed certificate created in step 10:
 - a. Click the **WAS LDAP self-signed certificate** check box.
 - b. Select **Extract**.
 - c. For Certificate file name, type a suitable file name.
 - d. For Data type, select **Base64-encoded ASCII data**.
 - e. Click **OK**.
12. Click **Save directly to the master configuration**.
13. Click **Save**.

The public key extracted in step 11 is copied to the LDAP server and imported into the LDAP server key database. This step allows the LDAP server to trust WebSphere Application Server and is necessary when client authentication is enabled on the LDAP server.

Using SSL between the Web server and WebSphere Application Server

SSL can be configured between the Web server and WebSphere Application Server to prevent attempts to intercept the data transmitted. Additionally, client authentication between the Web server and WebSphere Application Server can be used to limit WebSphere Application Server access to the Web server. Client authentication ensures that all requests to WebSphere Application Server must be routed through the environment as expected. It also helps to prevent attempts to bypass layers of security.

Enabling SSL

To enable SSL between WebSphere Application Server and the HTTP server:

1. Create a keystore.
2. Create a truststore.
3. Create and extract a self-signed certificate for WebSphere Application Server.
4. Create an SSL configuration.
5. Import the public certificate for the HTTP server.
6. Configure WebSphere Application Server to use the SSL configuration to communicate with the HTTP server.

Creating a keystore

To create a keystore:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **Key stores and certificates**.
5. Click **New**.
6. On the General Properties page:
 - a. For Name, type WebContainerKeyStore.
 - b. For Path, type `${CONFIG_ROOT}/cells/cellname/WebContainerKeyStore.p12` to place the keystore in the configuration directory of the deployment manager at a cell level. This path is synchronized with all nodes in the cell, enabling all nodes to access the keystore.
 - c. For Password and Confirm Password, type a suitable password, which is used to open the keystore.
 - d. For Type, select **PKCS12**.
 - e. Click **OK**.

Creating a truststore

To create a truststore:

1. Follow steps 1 through 5 on page 179 for creating a keystore.
2. Follow step 6 on page 179 to create a keystore. For the Name and Path, type `WebContainerTrustStore`, not `WebContainerKeyStore`.

Creating and extracting a self-signed certificate

To create and extract a self-signed certificate for WebSphere Application Server:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **Key stores and certificates**.
5. Select **WebContainerKeyStore**.
6. Select **Personal certificates**.
7. Select **Create a self-signed certificate**.
8. On the General Properties page:
 - a. For Alias, type WAS Web container self-signed certificate.
 - b. From the Key size list, select **1024**.
 - c. For Common name, type the host name of your server.
 - d. For Validity period, type an appropriate validity period. This is number of days for which the certificate is valid. Secure communication is not possible after the certificate expires, and a new certificate must be generated.
 - e. Click **OK**.
9. Extract the self-signed certificate created in step 8:
 - a. Select **WAS Web container self-signed certificate**.
 - b. Select **Extract**.
 - c. Enter a suitable file name for **Certificate file name**.
 - d. Select **Base64-encoded ASCII data for Data type**.
 - e. Click **OK**.
 - f. Click **Save directly to the master configuration**.
 - g. Select **Save**.

The public key extracted in step 9 is copied to the Web server and imported into the Web server key database so that the Web server can trust WebSphere Application Server.

Creating an SSL configuration

The next step is to create an SSL configuration. The SSL configuration is used to determine which truststore and keystore to use for creating the SSL connection.

In this example, a separate SSL configuration is created for SSL communication between WebSphere Application Server and the Web server.

To create an SSL configuration:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **SSL configurations**.
5. Click **New** to create an SSL configuration at the cell scope.
6. On the General Properties page:
 - a. For Name, type `WebContainerSSLSettings`.
 - b. For Trust store name, select **WebContainerTrustStore**.
 - c. For Key store name, select **WebContainerKeyStore**.
 - d. Select **Get Certificate aliases**.
 - e. Click **OK**.
7. Select **Quality of protection (QoP) settings**.
8. On the General Properties page:
 - a. For Client authentication, select **Required**.
 - b. Click **OK**.
9. Click **Save directly to the master configuration**.

Steps 7 and 8: Steps 7 and 8 are necessary only if you want to force clients to authenticate with WebSphere Application Server. This action is typically used to ensure that only the Web servers can communicate with the Web container in WebSphere Application Server. If you choose to do this, you must import the Web server certificate into the *WebContainerTrustStore*.

Importing the Web server certificate

The Web server certificate can be imported in one of the following ways:

- ▶ Import a certificate from a file
- ▶ Retrieve a certificate from a port

Importing a certificate from a file

To import a certificate from a file:

1. In the administrative console, navigate to **Key stores and certificates**.
2. Click **WebContainerTrustStore** (the truststore that you created earlier).
3. Select **Signer certificates**.
4. Click **Add**.

5. On the General Properties page:
 - a. For Alias, type Web Server Public Key.
 - b. For Filename, type `${CONFIG_ROOT}/HTTPPublicKey.arm`.
 - c. From Data type, select **Base64-encoded ASCII data**.
 - d. Click **OK**.
6. Click **Save directly to the master configuration**.

Retrieving a certificate from a port

To retrieve a certificate from a port:

1. In the administrative console, navigate to **Key stores and certificates**.
2. Click **WebContainerTrustStore** (the truststore that you created earlier).
3. Select **Signer certificates**.
4. Select **Retrieve from port**.
5. For Host, type the host name of the Web server.
6. For Port, type the HTTPS port. This is the SSL-enabled port on the Web server and is usually port 443.
7. For Alias, type Web Server Public Key.
8. Select **Retrieve signer information**.
9. Verify that the retrieve signer information is correct.
10. Click **OK**.
11. Click **Save directly to the master configuration**.

The final step is to configure WebSphere Application Server to use this new SSL configuration for communication with the Web server. To configure WebSphere Application Server:

1. Access the administrative console on the deployment manager.
2. Select **Security**.
3. Select **SSL certificate and key management**.
4. Select **Manage endpoint security configurations**.
5. Expand the Local Topology for Inbound → **cellName** → **nodes** → **nodeName** → **servers** → **serverName**.
6. Select **WC_defaulthost_secure**.
7. Select **Override inherited values**.
8. From SSL configuration, select **WebContainerSSLSettings**.
9. Select **Update certificate alias list**.

10. Click **OK**.
11. Repeat steps 5 through 9 for each application server on each node that you want to secure.
12. Click **Save directly to the master configuration**.
13. Click **Save**.

Application deployment

The following task maps users and groups contained in the LDAP user repository to the application security roles used in the LGI application. From the WebSphere Application Server 6.1 administrative console:

1. Select **Applications** and then **Enterprise Applications**.
2. For the LGI Web application, select **LGDSecureQuoteEAR**.
3. Select **Security role to user/group mapping**.
4. Select **Staff Discount Role** and select **Look up groups**.
5. Use search to navigate to the group representing LGI employees. Ensure that the group is displayed in the **Selected** column and click **OK**. This action restricts this role to users of this group.
6. For the roles All Quote Users and EJB Quote Users, select **All authenticated?**, which allows all users who are currently logged on to perform this role.
7. Click **OK**.
8. Restart the application for the changes to take effect.

Instructions for 3.5, 'WebSphere MQ on AIX and Linux for System z'

This section contains detailed instructions to configure WebSphere MQ on AIX or Linux for System z. WebSphere MQ 6.0 is similar on the two different platforms or any UNIX-based platform. The product is installed, configured, and administered in much the same way. Therefore, the information in this section applies to both platforms unless explicitly stated otherwise. For a high level view of this information, see 3.5, "WebSphere MQ on AIX and Linux for System z" on page 70.

Configuring security

This section provides detailed information for configuring WebSphere MQ access control on AIX or Linux for System z to meet the requirements of the LGI scenario.

Configuring access control

The following task shows how to provide sufficient access to the required WebSphere MQ AIX or Linux for System z queue manager resources for the LGI Web application. Note the following details:

- ▶ The WebSphere MQ Administrator user ID is `mqm`.
- ▶ The WebSphere Application Server process user who is currently logged on is `wasusr`.
- ▶ The AIX or Linux for System z operating system group containing `wasusr` is `wasgrp`.
- ▶ The local (AIX or Linux for System z) queue manager name is `LGI.WAS.Z.01`.
- ▶ The local queue accessed by the Web application's Java Messaging Service (JMS) receive method to obtain reply messages is `WAS.QUOTE.REPLY`.

To configure access control:

1. Log on to an operating system command line session as the WebSphere MQ Administrator:

```
su - mqm
```

2. Grant connect and inquire access to the queue manager for the group:

```
setmqaut -m LGI.WAS.Z.01 -t qmgr -g wasgrp +connect +inq
```

3. Grant put access to the cluster transmission queue for the group, allowing the WebSphere Application Server application to put messages to request queues on remote cluster queue managers:

```
setmqaut -m LGI.WAS.Z.01 -n SYSTEM.CLUSTER.TRANSMIT.QUEUE -t q -g wasgrp +put
```

4. Grant get, inquire, and browse access to all local reply queues for the group, allowing the JMS receive to obtain the reply messages:

```
setmqaut -m LGI.WAS.Z.01 -n WAS.QUOTE.REPLY -t q -g wasgrp +get +inq +browse
```

5. Review access granted to the queue manager resources:

```
dmpmqaut -m LGI.WAS.Z.01 -t q
```


6. Review access to the queue manager resources granted to the group:

```
dmpmqaut -m LGI.WAS.Z.01 -g wasgrp
```
7. Refresh the queue manager's security configuration for the changes to take effect:

```
runmqsc LGI.WAS.Z.01
refresh security
end
```
8. Log off the WebSphere MQ Administrator.

Debugging access control problems

To run a WebSphere MQ trace on AIX:

1. Log on to an AIX session as the WebSphere MQ Administrator:

```
su - mqm
```
2. Start WebSphere MQ trace:

```
trace -a -j30D,30E -o /tmp/trace.file
```
3. Recreate the access control problem.
4. Stop WebSphere MQ trace:

```
trcstop
```
5. Format WebSphere MQ trace output as follows, where /usr/mqm is the directory in which WebSphere MQ is installed:

```
/usr/bin/trcrpt -t /usr/mqm/lib/amqtrc.fmt /tmp/trace.file > /tmp/format.file
```
6. Open the formatted trace file and search for the phrase "insufficient authority" (associated with the MQOPEN). For example, the formatted trace file contains the following information:

```
30E 21.396561423 0.001862 Entity wasusr has
insufficient authority to access object WAS.QUOTE.REPLY
30E 21.396563114 0.001691 The following requested
permissions are unauthorized: browse
```

In this example, the group to which the user wasusr belongs requires browse access to the queue WAS.QUOTE.REPLY.
7. Log off the WebSphere MQ Administrator.

Configuring Secure Sockets Layer

The following task shows the steps that are required on an UNIX queue manager to enable SSL on sender and receiver cluster channels to and from a z/OS queue manager. Note the following details:

- ▶ The WebSphere MQ Administrator user ID is mqm.
- ▶ The local (UNIX) queue manager name is LGI.WAS.Z.01.
- ▶ The remote (z/OS) queue manager name is ST03.
- ▶ The host name of the z/OS machine is z0SHostname.

To configure SSL:

1. Obtain the public certificate for the remote (z/OS) queue manager. For example, the ST03 queue manager's certificate has been exported to a z/OS sequential file (called 'LGI.SSLKEY.ST03') as a Base64-encoded ASCII data certificate. To complete this task, open an operating system command line session and enter the following commands:

```
cd/tmp
ftp z0SHostname
ascii
quote site recfm=vb
get 'LGI.SSLKEY.ST03' ST03.crt
bye
```

Change the permissions on *ST03.crt* so that it can be read by the WebSphere MQ Administrator ID.

2. Obtain the value of the queue manager's property SSL key repository (SSLKEYR). On distributed platforms, this is set to `/var/mqm/qmgrs/QMgrName/ssl/key` by default, although the key repository does not exist until explicitly created. Consider the following example:
`/var/mqm/qmgrs/LGI!WAS!Z!01/ssl/key`

Periods in the queue manager name: If the queue manager name contains periods (for example, LGI.WAS.Z.01), the periods are replaced with exclamation marks in the file system directory (for example, LGI!WAS!Z!01).

3. Open the key management tool, in preparation for creating and managing key database file and certificates.

Key management tool: The following instructions and window captures are from the **gsk7ikm** key management tool. Alternative key management tools are available, for example **ikeman**, which provide the same functionality, through slightly different interfaces.

On AIX, the **gsk7ikm** tool was used in the LGI scenario. On Linux for System z, the **ikeman** tool was used in the LGI scenario.

- a. Log on to an operating system command line session as the WebSphere MQ Administrator:

```
su - mqm
```

- b. Set up the local environment to run Global Security Kit (GSKit) key management tool GUI **gsk7ikm** by entering the following commands:

- i. Export DISPLAY to a machine with X11 support:

```
Export DISPLAY=hostname:0
```

- ii. Export JAVA_HOME to the jre directory that is installed as part of WebSphere MQ:

```
Export JAVA_HOME=/usr/mqm/ssl/jre
```

- iii. Open the key management GUI:

```
/usr/bin/gsk7ikm
```

Figure B-2 shows the GUI for the **gsk7ikm** key management tool.

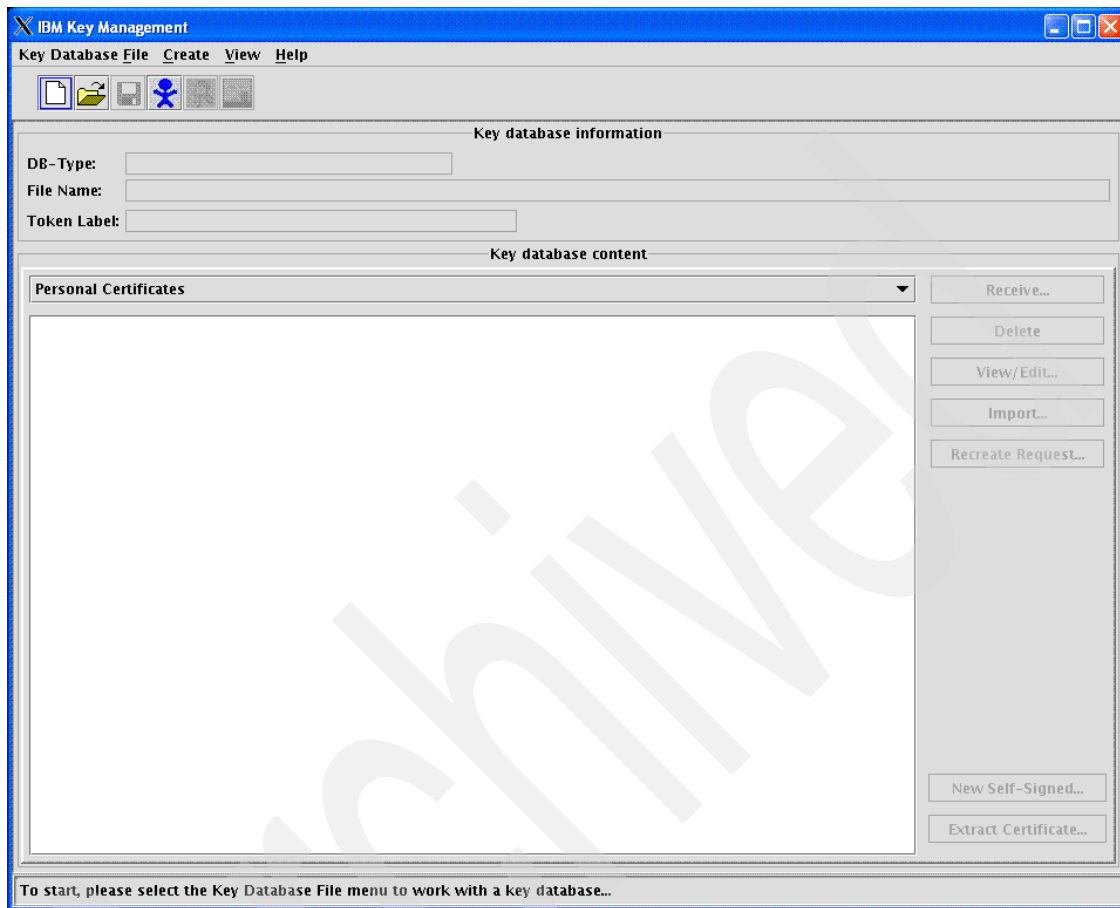


Figure B-2 Key management tool **gsk7ikm** user interface

4. Create a new key database (using **gsk7ikm**):
 - a. Select **Key Database File**.
 - b. Click **New**.

- c. In the New window (Figure B-3):
 - i. For Key database type, select **CMS**.
 - ii. For File Name, type `key.kdb`, which is the value of the file name specified on the queue manager's `SSLKEYR` property.
 - iii. For Location, type `/var/mqm/qmgrs/LGI!WAS!Z!01/ssl`, which is the value of the path specified on the queue manager's `SSLKEYR` property.
 - iv. Click **OK**.

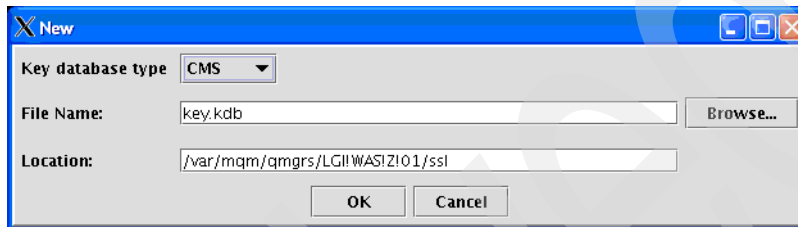


Figure B-3 Key management tool `gsk7ikm`, creating key database

- d. In the next window, enter a password for the keystore and select the **stash password to a file** option. Then click **OK**.
- 5. Create a self-signed certificate for the local queue manager in the key database (using `gsk7ikm`):
 - a. From the pull-down option, select **Personal Certificates**.
 - b. Select **New Self-Signed**.

- c. In the Create New Self-Signed Certificate window (Figure B-4):
 - i. For the Key Label, type `ibmwebspheremqlgi.was.z.01`.
 - ii. For Version, accept the default value of **X509 V3**.
 - iii. For Key Size, accept the default value of **1024**.
 - iv. For Common Name, type `LGI.WAS.Z.01`.
 - v. For Country or region, type `GB`.
 - vi. Optionally enter values for the other fields.
 - vii. Click **OK**.

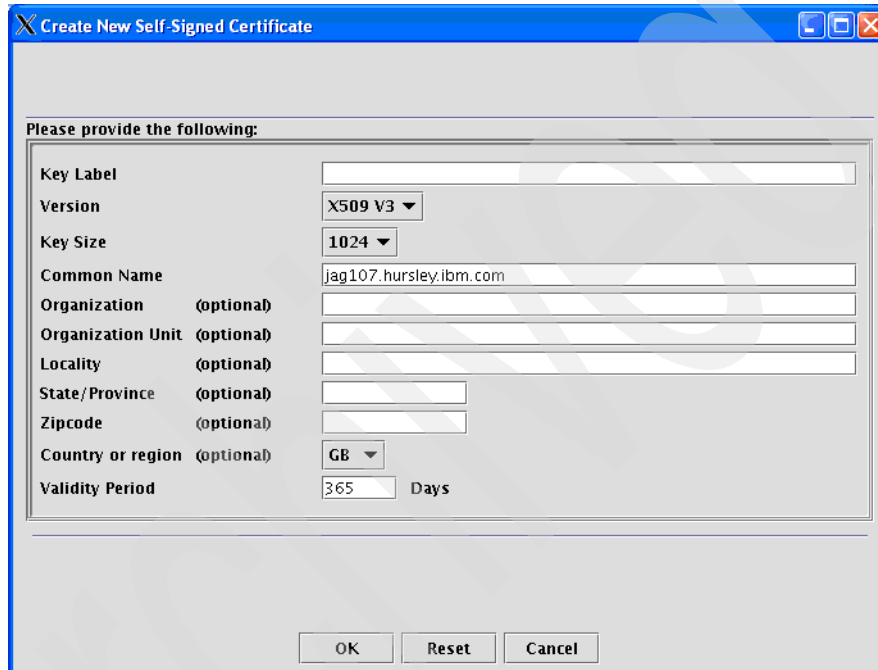


Figure B-4 Creating a self-signed certificate by using key management tool `gsk7ikm`

6. Extract the local queue manager's public certificate so that it can be added later into the remote queue manager's key ring (using `gsk7ikm`):
 - a. Select **Extract Certificate**.
 - b. For Data Type, select **Base64-encoded ASCII data**.
 - c. For Certificate file name, type `LGIWASZ01.arm`.
 - d. For Location, type `/tmp`.

7. Add the remote (z/OS) queue manager's public certificate to the key database (using **gsk7ikm**):
 - a. Change the pull-down to **Signer Certificates**.
 - b. Select all existing default certificates and select **Delete**. This step restricts access to this queue manager to only those clients whose certificates are explicitly added to this key database.
 - c. Click **Add**.
 - d. For Data Type, select **Base64-encoded ASCII data**.
 - e. Enter the location and name of the public certificate obtained from the z/OS queue manager (for example, /tmp/ST03.crt).
 - f. For the label, type `ibmWebSphereMQST03`.
8. Set the cipher specification on the sender channel to match the value specified on the equivalent receiver channel on the remote (z/OS) queue manager:
 - a. Log on to an operating system command line session as the WebSphere MQ Administrator by entering the following commands:


```
su - mqm
runmqsc LGI.WAS.Z.01
ALTER CHANNEL CHANNEL('senderChannelName') CHLTYPE(CLUSSDR)
SSLCIPH(RC4_MD5_US)
```
 - b. Log off the WebSphere MQ Administrator.
9. Set cipher specification to be used on receiver channel and request client authentication:
 - a. Log on to an operating system command line session as the WebSphere MQ Administrator by entering the following commands:


```
su - mqm
runmqsc LGI.WAS.Z.01
ALTER CHANNEL CHANNEL('receiverChannelName') CHLTYPE(CLUSRCVR)
SSLCIPH(RC4_MD5_US) SSLCAUTH(REQUIRED)
```
 - b. Log off the WebSphere MQ Administrator.
10. Refresh the queue manager's security settings to implement the changes:
 - a. Log on to an operating system command line session as the WebSphere MQ Administrator by entering the following commands:


```
su - mqm
runmqsc LGI.WAS.Z.01
refresh security
end
```
 - b. Log off the MQ Administrator.

11. Close the key database file and exit from **gsk7ikm**. Log off the WebSphere MQ Administrator.
12. Restart both ends of each cluster sender and receiver channel for SSL settings to be implemented, including the dynamic cluster sender channels. These are defined automatically with the same SSLCIPH value as the corresponding receiver channel after they are restarted.

Troubleshooting

Error messages can occur after configuring SSL for WebSphere MQ and suggests possible resolutions. One error message is “AMQ9660: SSL key repository: password stash file absent or unusable.”

To solve this problem, the key repository password stash file is created during creation of a key database. This file has a file type extension of .sth and exists in the same directory as the key database .kdb file. Check the value specified for the queue manager's property SSL key repository (SSLKEYR) and ensure the following items:

- ▶ The value only specifies the path and file name of the key database file and not the file type extension.
- ▶ The path and name are correct (list contents of the directory specified by the path on the file system). For example, if your key database file is called myKey.kdb in the /var/mqm/qmgrs/myQmgr/ssl directory, then the /var/mqm/qmgrs/myQmgr/ssl directory should contain these four files: myKey.crl, myKey.kdb, myKey.rdb, and myKey.sth. Also the SSLKEYR property value should be /var/mqm/qmgrs/myQmgr/ssl myKey.
- ▶ The queue manager has read access to all the files that are associated with the key database and the directory that contains them.

Instructions for 3.6, ‘DB2 Enterprise 9 on AIX and Linux for System z’

This section contains detailed instructions to install and configure DB2 Enterprise 9 on AIX or Linux for System z. DB2 is similar on the two different platforms or any UNIX-based platform. The product is installed, configured, and administered in much the same way. Therefore, the information applies to both platforms unless explicitly stated otherwise. For a high level view of this information, see 3.6, “DB2 Enterprise 9 on AIX and Linux for System z” on page 76.

Database definition

In the LGI scenario environment each database is created in its own file system. This section provides detailed information about the steps to create such a file system on AIX.

Creating a file system

The steps in this task will vary slightly depending on the hardware used. The following instructions are based on an IBM eServer™ pSeries® 5 server.

To create a file system:

1. Enter the following command:
`smitty storage`
2. Select **Logical Volume Manager** and click **Enter**.
3. Select **Logical Volumes** and click **Enter**.
4. Select **Add a Logical Volume** and click **Enter**.
5. For the volume group name, type `sharkvg`, where a volume group exists on hardware that can be accessed and is suitable for the database being created. Click **Enter**.
6. For the new logical volume name, type `logvol1`.
7. For the number of logical partitions for the size of database required, type `8`.
8. For the physical volume name, type `hdisk1`.
9. For the logical volume type, enter `jfs`.
10. For the maximum number of physical volumes, type `1`.
11. Click **Enter**.
12. Press the PF3 keyboard key several times until you return to the Systems Storage Management page.
13. Select **File Systems** and click **Enter**.
14. Select **Add / Change / Show / Delete File Systems** and click **Enter**.
15. Select **Journaled File Systems** and click **Enter**.
16. Select **Add a Journaled File System on a Previously Defined Logical Volume** and click **Enter**.
17. Select **Add a Standard Journaled File System** and click **Enter**.
18. Press the PF4 keyboard key and select the logical volume, **logvol1**, which was created previously. Click **Enter**.
19. For the mount point, type `/database1`.

20. For Mount AUTOMATICALLY at system restart?, select **Yes** and click **Enter**.
21. Press the PF3 key on your keyboard several times to leave **smitty**.
22. Enter the following command:

```
mount /database1
```
23. Enter the following command substituting the instance owner, instance group, and database mount point (file system):

```
chown db2inst2:db2grp2 database1
```

Instructions for 3.7, 'IBM HTTP Server 6.1 on AIX'

This section contains detailed instructions to configure the HTTP server. For a high level view of this information, see 3.7, "IBM HTTP Server 6.1 on AIX" on page 78.

Configuring security

This section explains the detailed steps to configure IBM HTTP Server to use SSL (HTTPS) on its connections with the browsers. In the following task, we used self-signed certificates. However, in a production system, obtain appropriate certificates from a certificate authority.

Enabling HTTPS communication with browsers

First, create a CMS key containing a self-signed certificate:

1. Open the key management GUI:

```
/usr/bin/gsk7ikm
```
2. Select **Key Database File** and click **New**.
3. On the next page:
 - a. For Key database type, select **CMS**.
 - b. For File Name, type `key.kdb`.
 - c. For Location, type `/usr/IBM/HTTPServer/keys/`.
 - d. Click **OK**.
 - e. Enter a password for the key database and select the **Stash password to a file** option.
 - f. Click **OK**.

4. Create a self-signed certificate for IBM HTTP Server in the key database:
 - a. Select **Personal Certificates** from the list and select **New Self-Signed**.
 - b. For the Key Label, type Web Server SSL Certificate.
 - c. For Version, accept the default value of **X509 V3**.
 - d. For Key Size, accept the default value of **1024**.
 - e. For Common Name, type the Web server host name.
 - f. For Country or region, type GB.
 - g. Optionally enter values for the other fields.
 - h. Click **OK**.

Then, enable HTTPS:

1. Add the following entry to the `httpd.conf` file to configure the Web server to listen on port 443:

```
Listen *:443
```

2. Add the following entries to the `httpd.conf` file to configure the Web server to use SSL on port 443 by using the CMS database that was created previously:

```
<VirtualHost *:443>
SSLEnable
KeyFile "/usr/IBM/HTTPServer/keys/key.kdb"
</VirtualHost>
SSLDisable
```

3. Restart the HTTP server.

Instructions for 3.9, 'WebSphere MQ on z/OS'

This section contains detailed instructions to configure WebSphere MQ on z/OS. For a higher level view of this information, see 3.9, "WebSphere MQ on z/OS" on page 88.

Configuring access control

The following task shows how to provide sufficient access to the required WebSphere MQ z/OS queue manager resources for the LGI Web application. Note the following details:

- ▶ The WebSphere MQ Administrators user ID is part of the WebSphere MQADMIN RACF group.
- ▶ The system tasks IDs for the queue managers and channel initiators belong to the RACF group WebSphere MQTASKS.

- ▶ The RACF group in which the WebSphere Application Server cell system task IDs currently belong is PSA2SRG.
- ▶ The local (z/OS) queue manager name is CSQ1 is a member of the queue-sharing group MQGT.
- ▶ The local queue that is accessed by the Web application's JMS receive method to obtain reply messages is WAS.QUOTE.REPLY.

To configure access control:

1. Log on to a Time Sharing Option (TSO) session with a user ID that is part of the WebSphere MQADMIN RACF group.
2. Define the RACF profile and grant update access to all local reply and request queues for the group to allow the JMS receive to obtain the reply messages. Enter the following RACF commands:

```
RDEF MQQUEUE MQGT.WAS.QUOTE.REPLY UACC(NONE) OWNER(WebSphere
MQADMIN)
PE MQGT.WAS.QUOTE.REPLY CL(MQQUEUE) ID(WebSphere MQADMIN)
ACC(UPDATE)
PE MQGT.WAS.QUOTE.REPLY CL(MQQUEUE) ID(WebSphere MQTASKS)
ACC(UPDATE)
PE MQGT.WAS.QUOTE.REPLY CL(MQQUEUE) ID(PSA2SRG) ACC(UPDATE)
SETR GENERIC(MQQUEUE) RACLIST(MQQUEUE) REFRESH
```

3. Refresh the queue manager's security configuration to implement the changes addressed by the commands below:
 - a. Enter the following MQM command:


```
REFRESH SECURITY TYPE(MQQUEUE) CMDSCOPE(*)
```
 - b. Enter the following MQM command:


```
RVERIFY SECURITY(WAS_USER_ID)
```
4. Log off the WebSphere MQ Administrator.

Configuring Secure Sockets Layer

The following task shows how to enable a z/OS queue manager to use SSL and to create a self-signed certificate for the queue manager. This task does not use the Integrated Cryptographic Service Facility (ICSF) to store certificates. If ICSF is used, extra setup is required.

In this example, note the following details:

- ▶ The channel initiator address space user ID is SYSTASK.
- ▶ The local (z/OS) queue manager name is CSQ1.
- ▶ The z/OS queue manager's key ring is WebSphere MQCSQ1KEYRING.

To configure SSL:

1. Grant the channel initiator address space user ID access to RACF key rings:

```
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(SYSTASK)
ACCESS(READ)
```

Restriction: If the queue manager's key ring is not owned by the channel initiator address space user ID, ACCESS(UPDATE) is required in this command.

2. Enable the queue managers to use SSL by entering the following MQM command. The value must be greater than one but less than 50 to prevent storage allocation problems.

```
ALTER QMGR SSLTASKS(2)
```

3. Restart the channel initiator address space (CHINIT) to implement the change.

4. Create the key ring for the queue manager:

```
RACDCERT ID(SYSTASK) ADDRING(WebSphere MQCSQ1KEYRING)
```

5. Add the key ring to the queue manager by setting the queue manager's property SSL key repository (SSLKEYR) with the following WebSphere MQ command:

```
ALTER QMGR SSLKEYR(WebSphere MQCSQ1KEYRING)
```

6. Create a self-signed certificate for the local queue manager:

```
RACDCERT ID(SYSTASK) GENCERT SUBJECTSDN(CN('CSQ1CN') O('LGI')
C('GB') T('CSQ1 CERTIFICATE')) WITHLABEL('ibmWebSphereMQCSQ1')
```

7. Add the self-signed certificate to queue manager's key ring:

```
RACDCERT ID(SYSTASK) CONNECT(ID(SYSTASK) LABEL('ibmWebSphereMQCSQ1')
RING(WebSphere MQCSQ1KEYRING) USAGE(PERSONAL))
```

8. Enter the following RACF commands to apply the changes:

```
SETROPTS RACLIST (DIGTCERT) REFRESH
SETROPTS RACLIST (DIGTRING) REFRESH
```

9. Refresh the queue manager's SSL settings to implement key ring changes by entering the following MQM command:

```
REFRESH SECURITY TYPE(SSL)
```

Exchanging certificates with another z/OS queue manager

The following task shows the steps on a z/OS queue manager to enable SSL on sender and receiver cluster channels to and from another z/OS queue manager.

This task includes the following details:

- ▶ The local (z/OS) queue manager name is CSQ1.
- ▶ The channel initiator address space user ID of CSQ1 is SYSTASK1.
- ▶ The key ring for CSQ1 is WebSphere MQCSQ1KEYRING.
- ▶ One of the remote (z/OS) queue managers is named CSQ2.
- ▶ The channel initiator address space user ID of CSQ2 is SYSTASK2.
- ▶ The label of CSQ2's self signed certificate is *ibmWebSphereMQCSQ2*.

Note the following considerations:

- ▶ RACF commands are issued from the Interactive System Productivity Facility (ISPF) TSO command interface, while the user is logged on with sufficient RACF authority to create key rings and certificates.
- ▶ MQM commands are issued from the ISPF MQ utilities interface.

To exchange certificates with another z/OS queue manager:

1. Connect the remote (z/OS) queue manager's public certificate to local queue manager's key ring:

```
RACDCERT ID(SYSTASK1) CO (ID(SYSTASK2) LABEL('ibmWebSphereMQCSQ2')  
-RING(WebSphere MQCSQ1KEYRING) USAGE(personal))
```

2. Repeat the previous step with all other queue managers that will be part of this cluster until the local queue manager is connected to all the self-signed certificates of the other queue managers.

3. Enter the following RACF commands to pick up the changes:

```
SETROPTS RACLIST (DIGTCERT) REFRESH  
SETROPTS RACLIST (DIGTRING) REFRESH
```

4. Refresh the queue manager's SSL settings to implement key ring changes by entering the following MQM command:

```
REFRESH SECURITY TYPE(SSL)
```

5. Set the cipher specification on the sender channel to match the value specified on the equivalent receiver channel on the remote queue manager:

```
ALTER CHANNEL('senderChannelName') CHLTYPE(CLUSSDR)  
SSLCIPH(RC4_MD5_US)
```

6. Set the cipher specification to be used on the receiver channel and request client authentication:

```
ALTER CHANNEL('receiverChannelName') CHLTYPE(CLUSRCVR)
SSLCIPH(RC4_MD5_US) SSLCAUTH(REQUIRED)
```
7. Restart both ends of each cluster sender and receiver channel to implement the SSL settings. This restart includes dynamic cluster sender channels. The dynamic cluster sender channels should be defined automatically with the same SSLCIPH value as the corresponding receiver channel after they are restarted.

Instructions for 3.11, 'WebSphere MQ File Transfer Edition on AIX'

This section contains detailed instructions to configure WebSphere MQ File Transfer Edition (FTE) on AIX. For a high level view of this information, see 3.11, "WebSphere MQ File Transfer Edition on AIX" on page 94.

Configuring an AIX agent

To configure a WebSphere MQ FTE agent:

1. Install WebSphere MQ FTE. Choose the server installation option so that bindings connections to queue managers are supported.
2. Before creating a WebSphere MQ FTE agent, ensure that a coordination queue manager is available. See the back-end WebSphere MQ FTE section to install and configure the coordination queue manager.
3. Optional: Set the PATH environment variable to include the WebSphere MQ FTE bin directory:

```
export PATH=$PATH:/usr/fte/bin
```

4. Set the FTE_JAVA_HOME environment variable to the Java 5 installation path:

```
export FTE_JAVA_HOME=/usr/java5
```

5. Set the FTE_CONFIG environment variable to the WebSphere MQ FTE data directory:

```
export FTE_CONFIG=/var/fte
```

This directory contains all configuration and status information for the WebSphere MQ FTE installation.

6. Create the files that are used to connect to the coordination queue manager. Use the following command to create the data directory (for example, /var/fte), the coordination queue manager subdirectory (for example, /var/fte/LGI.BACK.FTE.CQM), the wmqfte.properties file, and the coordination.properties file:

```
fteSetupCoordination -coordinationQMGr LGI.BACK.FTE.CQM
```

7. Create the files that are used to connect to the command queue manager:

```
fteSetupCommands -connectionQMGr LGI.FRONT.AIX.01
```

In this case, the command queue manager is the local queue manager to which the agent will connect. This will create the command.properties file.

8. Create an agent, which creates an agent subdirectory (for example, /var/fte/LGI.BACK.FTE.CQM/agents/AIX01) that includes the AIX01_create.mqsc file:

```
fteCreateAgent -agentName AIX01 -agentQMGr LGI.FRONT.AIX.01
```

9. Create the queues that are required by the agent. Use the AIX01_create.mqsc file as follows:

```
cat AIX01_create.mqsc | runmqsc LGI.FRONT.AIX.01
```

10. Start the agent as a background process:

```
fteStartAgent -F AIX01
```

11. Check that the agent is running:

```
fteListAgents
```

12. If the agent is not listed, check the agent log for errors and check for first failure data captures (FFDCs). Both the log file and any FFDCs are in agent logs directory (for example, /var/fte/LGI.BACK.FTE.CQM/agents/AIX01/logs). Also check the status of the WebSphere MQ channels between the agent queue manager and the coordination queue manager and the WebSphere MQ error logs.

Stopping an agent: In addition to starting and listing agents, you can stop agents by using the **fteStopAgent** command.

Transferring files

File transfers are scheduled so that files in the /var/portal/claim_jpgs directory are transferred from the front-end machines to the back-end machine. The following example shows how the command to setup the transfer from the first front-end machine to the back-end machine:


```
fteCreateTransfer -sa AIX01 -sm LGI.FRONT.AIX.01 -da AA -dm BE01 -dd  
/claims/data/claim_jpgs/ -de overwrite -sd delete -t binary -oi hours  
-of 1 /var/portal/claim_jpgs/*
```

Although the files are stored in the UNIX System Services file system on BE01, WebSphere MQ FTE supports writing to partitioned data sets (PDSs).

Instructions for 3.12, ‘IBM Tivoli Composite Application Manager on AIX’

This section contains detailed instructions to configure IBM Tivoli Composite Application Manager on AIX. For a high level view of this information, see 3.12, “IBM Tivoli Composite Application Manager on AIX” on page 96. You must configure a Tivoli Enterprise Management Agent and IBM Tivoli Composite Application Manager for WebSphere Data Collector on every front-end machine that you want to monitor.

Configuring Tivoli Enterprise Management Agent

To configure Tivoli Enterprise Management Agent:

1. Before installing Tivoli Enterprise Management Agent or WebSphere Data Collector, ensure that the back-end IBM Tivoli Monitoring environment is configured. See 4.11, “IBM Tivoli Monitoring on AIX” on page 134.
2. Install Tivoli Enterprise Management Agent by using the instructions as explained in the “Installing the Monitoring Agent on UNIX and Linux” topic in the Tivoli Composite Application Manager for WebSphere Information Center at the following Web address:

http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/topic/com.ibm.itcamwas.doc_6.1/KYN61IG72.htm#wq79

Configuring WebSphere Data Collector

To configure WebSphere Data Collector:

1. Before installing WebSphere Data Collector:
 - a. Install and configure IBM Tivoli Monitoring. See 4.11, “IBM Tivoli Monitoring on AIX” on page 134.
 - b. Install and configure Tivoli Enterprise Management Agent.

2. Install WebSphere Data Collector by following the instructions in “Data Collector on UNIX and Linux” in the Tivoli Composite Application Manager Information Center at the following address:
http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.itcamwas.doc_6.1/itcam_61_4_was_dc_dist_install_guide118.htm
3. Download and install IBM Tivoli Composite Application Manager for WebSphere Data Collector 6.1 iFix (12) as explained in “ITCAM for WebSphere 6.1 FP3, Interim Fix 12, 6.1.0.3-TIV-ITCAMfWAS_MP-IF0012,” at the following address:
<http://www-01.ibm.com/support/docview.wss?&uid=swg24020453>

Note: Ensure that IBM Tivoli Composite Application Manager for WebSphere Data Collector 6.1 Fix Pack 4 is installed rather than iFix 12, if available.

Verifying the installation by using Tivoli Enterprise Portal

The installation can be verified by logging on to the Tivoli Enterprise Portal. The following task explains how to log on to the Tivoli Enterprise Portal by using the browser client from Microsoft Internet Explorer® v5 or later, as recommended by the IBM Tivoli Monitoring documentation:

1. Open Internet Explorer v5 or later and navigate to the following URL:
`http://<hostname>:1920///cnp/client`
2. Check to see that the server, WebSphere Agent, WebSphere Application Server profile, and related data are available.

Figure B-5 shows the GUI for Tivoli Enterprise Portal.

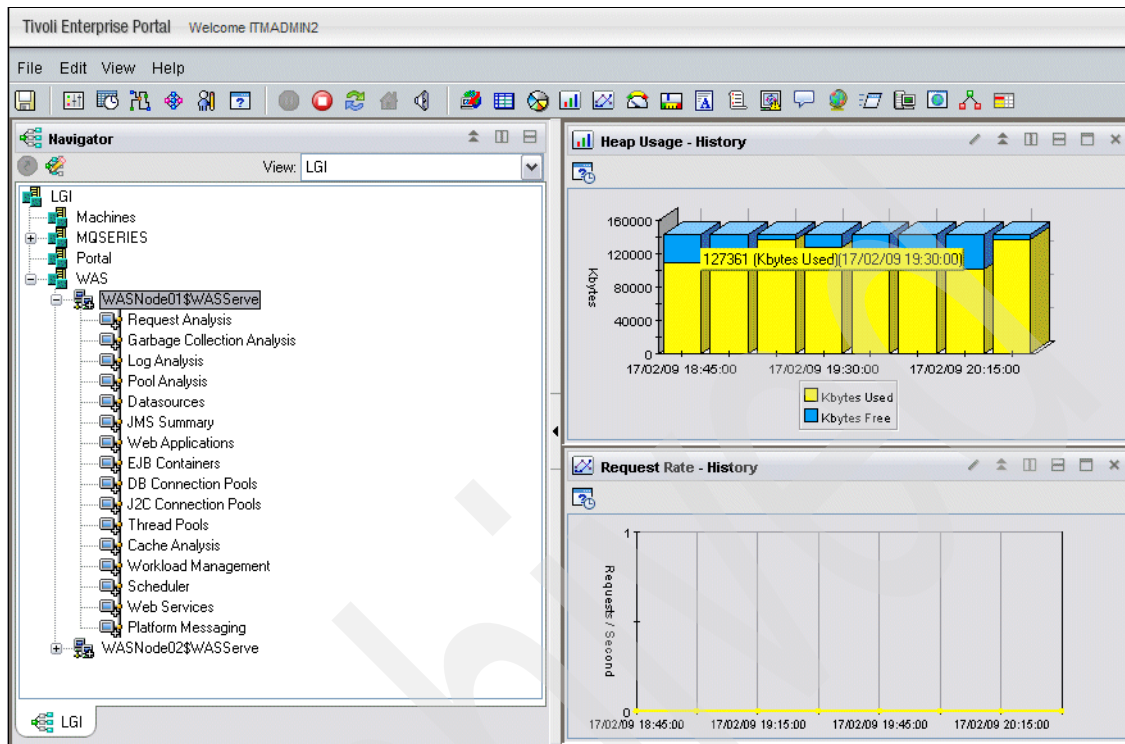


Figure B-5 User interface for the Tivoli Enterprise Portal

Note the following points:

- ▶ If the WebSphere Agent is not shown, then application support was not added to Tivoli Enterprise Portal Server correctly.
- ▶ If the WebSphere Application Server profile and related data are not shown, check the WebSphere server logs. There should be messages that contain "CYN" soon after the server starts.
- ▶ Check the DC log file, for example, `/var/ibm/tivoli/common/CYN/logs/PortalNode1.WebSphere_Portal/trace-dc-native.log`.

Instructions for 4.3, ‘WebSphere Process Server on z/OS’

This section contains detailed instructions to configure WebSphere Process Server on z/OS. For a higher level view of this information, see 4.3, “WebSphere Process Server on z/OS” on page 103.

Configuring security

The following tasks shows how to enable SSL on a WebSphere MQ link sender and receiver cluster for WebSphere Process Server to a WebSphere MQ queue manager. Note the following details:

- ▶ The SSL configuration for the WebSphere MQ link is `BPCBusSSLConfig`.
- ▶ The keystore for the WebSphere MQ link SSL configuration is `BPCBusKeyStore`.
- ▶ The truststore for the WebSphere MQ link SSL configuration is `BPCBusTrustStore`.
- ▶ The certificate alias for the WebSphere MQ link is `bpccertificate`.
- ▶ The filename of the (binary DER) certificate exported from the WebSphere MQ queue manager is `wmqcert.der`.
- ▶ The WebSphere MQ certificate label (WebSphere MQ certificate labels must conform to strict naming conventions) is `ibmWebSphereMQQMGR`.
- ▶ The BPC service integration bus is `BPC.cellName.Bus`.
- ▶ The SCA system service integration bus is `SCA.SYSTEM.cellName.Bus`.
- ▶ The foreign bus name of the MQ link is `MQLinkForeignBus`.
- ▶ The WebSphere MQ link is `MQLinkname`.
- ▶ The receiver channel on the WebSphere MQ link is `WebSphere MQ.TO.BPC`.
- ▶ The sender channel on the WebSphere MQ link is `BPC.TO.WebSphere MQ`.

To configure security:

1. Create a keystore at the cell scope to store the private key for the BPC service integration bus:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Key stores and certificates**.
 - e. Click **New**.

- f. For Name, type BPCBusKeyStore.
 - g. For Path, type \${CONFIG_ROOT}/cells/cellName/BPCBusKeyStore.p12.
 - h. For Password and Confirm Password, enter a valid password.
 - i. For Type, accept the default value of **PKCS12**.
 - j. Click **OK**.
2. Create a self-signed certificate for the BPC service integration bus in the keystore:
- a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Key stores and certificates**.
 - e. Select **BPCBusKeyStore**.
 - f. Select **Personal Certificates**.
 - g. Select **Create a self-signed certificate**.
 - h. For Alias, type bpccertificate.
 - i. For Common Name, type the TCP/IP address of the host server where WebSphere Process Server is running.
 - j. For Organization, type LGI.
 - k. Optional: For Country or region, modify the default value of **US**.
 - l. Click **OK**.
 - m. Save the changes and ensure that they are synchronized with the nodes.
3. Extract the public part of the BPC service integration bus certificate to add it to the key ring of the WebSphere MQ queue manager:
- a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Key stores and certificates**.
 - e. Select **BPCBusKeyStore**.
 - f. Select **Personal Certificates**.
 - g. Select **bpccertificate** and select **Extract**.
 - h. For Certificate file name, type
\${CONFIG_ROOT}/cells/cellName/bpccertificate.der.
 - i. For Data type, select **Binary DER data**.
 - j. Click **OK**.

- k. Give the extracted BPC service integration bus certificate to the WebSphere MQ administrator so that it can be added to the WebSphere MQ queue manager's key ring.
4. Create a truststore to store the public key of the WebSphere MQ queue manager to which WebSphere MQ link connects:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Key stores and certificates**.
 - e. Select **New**.
 - f. For Name, type `BPCBusTrustStore`.
 - g. For Path, type `${CONFIG_ROOT}/cells/cellName/BPCBusTrustStore.p12`.
 - h. For the Password and Confirm Password, type a value.
 - i. For Type, accept the default value of **PKCS12**.
 - j. Click **OK**.
5. Add WebSphere MQ queue manager's public certificate to the truststore:
 - a. Obtain the public certificate of the WebSphere MQ queue manager that the WebSphere MQ link connects to from the WebSphere MQ Administrator.
 - b. Copy the certificate to the `{CONFIG_ROOT}/cells/cellName/directory`, which is one that is accessible to the WebSphere Process Server administrative console.
 - c. Log on to the WebSphere Process Server administrative console.
 - d. Select **Security**.
 - e. Select **SSL certificate and key management**.
 - f. Select **Key stores and certificates**.
 - g. Select **BPCBusTrustStore**.
 - h. Select **Signer Certificates**.
 - i. Click **Add**.
 - j. For Alias, type `ibmWebSphereMQQMGR`, where the alias must match the value specified for the label when the certificate was created.
 - k. For File name, type `CONFIG_ROOT}/cells/cellName/wmqcert.der`.
 - l. For Data Type, select **Binary DER data**.
 - m. Click **OK**.
 - n. Save the changes and ensure that they are synchronized with the nodes.

6. Create a new SSL configuration to reference the keystore and truststore:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **SSL configurations**.
 - e. Select **New JSSE configuration**.
 - f. For Name, type **BPCBusSSLConfig**.
 - g. For Trust store name, select **BPCBusTrustStore**.
 - h. For Key store name, select **BPCBusKeyStore**.
 - i. Select **Get certificate aliases**.
 - j. For Default server certificate alias, select **bpccertificate**.
 - k. For Default client certificate alias, select **bpccertificate**.
 - l. Click **Apply**.
 - m. Select **Quality of Protection (QoP) settings**.
 - n. For Client authentication, select **Required**.
 - o. Accept the default values for all other fields. By default, all supported ciphers are selected, including values compatible with WebSphere MQ.
 - p. Save the changes and ensure that they are synchronized with the nodes.
7. Associate the SSL configuration with the inbound WebSphere MQ link. Define this at the server scope:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Manage endpoint security configurations**.
 - e. Expand the Local Topology for **Inbound** → *cellName* → **nodes** → *nodeName* → **servers** → *serverName*.
 - f. Select **SIB_MQ_ENDPOINT_SECURE_ADDRESS**.
 - g. For Specific SSL configuration for this endpoint, select **Override inherited values**.
 - h. For SSL configuration, select **BPCBusSSLConfig**.
 - i. Select **Update Certificate Alias List**.
 - j. Click **OK**.

- k. Repeat these steps for any other servers that are using the WebSphere MQ link on the BPC service integration bus.
 - l. Save the changes and ensure that they are synchronized with the nodes.
8. Associate the SSL configuration with the outbound WebSphere MQ link. Define this at the server scope:
- a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **SSL certificate and key management**.
 - d. Select **Manage endpoint security configurations**.
 - e. Expand the Local Topology for **Outbound** → **cellName** → **nodes** → **nodeName** → **servers** → **serverName** → **Bus to WebSphere MQ**.
 - f. For Specific SSL configuration for this endpoint, select **Override inherited values**.
 - g. For SSL configuration, select **BPCBusSSLConfig**.
 - h. Select **Update Certificate Alias List**.
 - i. Click **OK**.
 - j. Repeat these steps for any other servers that are using the WebSphere MQ link on the BPC service integration bus.
 - k. Save the changes and ensure that they are synchronized with the nodes.
9. Configure the WebSphere MQ link sender channel to use SSL:
- a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **Bus Security**.
 - d. Select **BPC.cellName.Bus**.
 - e. For Topology, select **Foreign buses**.
 - f. Select **MLinkForeignBus**.
 - g. For Related items, select **WebSphere MQ link**.
 - h. For Additional properties, select **Sender channel**.
 - i. Select **BPC.TO.WebSphere MQ**.
 - j. For Transport chain, select **OutboundSecureMLink**.
 - k. Click **OK**.
 - l. Save the changes and ensure that they are synchronized with the nodes.

10. Enable security on the BPC service integration bus:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Security**.
 - c. Select **Bus Security**.
 - d. Select **BPC.cellName.Bus**.
 - e. For Additional properties, select **Security**.
 - f. For General properties, select **Enable bus security**.
 - g. For Permitted transports, select **Restrict the use of defined transport channel chains to the list of permitted transports**.
 - h. Click **OK**.
 - i. Save the changes and ensure that they changes are synchronized with the nodes.
11. Grant existing user authority to exchange messages between the BPC and SCA system service integration buses:
 - a. Log off the WebSphere Process Server administrative console to avoid any configuration clashes.
 - b. Invoke the administrative scripting program (wsadmin) from the WebSphere Process Server deployment manager's (dmgr) bin directory by entering the following commands on a UNIX System Service command line:

```
install_root/bin/wsadmin.sh -user wpsadmin -password  
wpsadminPassword -host dmgrhostname -port dmgrport  
$AdminTask addUserToForeignBusRole {-bus BPC.cellName.Bus bus1  
-foreignBus SCA.SYSTEM.cellName.Bus -role sender -user wpsadmin}  
$AdminConfig save
```
12. Stop and start the WebSphere MQ link channels to make the SSL configuration changes take effect:
 - a. Log on to the WebSphere Process Server administrative console.
 - b. Select **Service integration**.
 - c. Select **Buses**.
 - d. Select **BPC.cellName.Bus**.
 - e. For Topology, select **Foreign buses**.
 - f. Select **MLinkForeignBus**.
 - g. For Related items, select **WebSphere MQ link** to view the MQ link page.
 - h. For Additional properties, select **Receiver channel**.
 - i. For Quiesce state, select **Force**.

- j. For Target state, select **Stopped**.
- k. Select **WebSphere MQ.TO.BPC** and click **Stop**. Wait for the channel to stop.
- l. Select **WebSphere MQ.TO.BPC** and click **Start**. The channel should change to inactive state. The channel only changes to a *started* state when the WebSphere MQ queue manger sends a request.
- m. Return to the MQ link page.
- n. For Additional properties, select **Sender channel**.
- o. For Target state, select **Stopped**.
- p. Select **BPC.TO.WebSphere MQ** and click **Stop**. Wait for the channel to stop.
- q. Select **BPC.TO.WebSphere MQ** and click **Start**. The channel should change to a *started* state.

DER format and FTP mode: When exchanging certificates on z/OS, use the binary DER format and FTP binary mode to avoids any certificate corruption that might be caused by conversion problems between ASCII and EBDIC.

Instructions for 4.4, ‘WebSphere Service Registry and Repository on z/OS’

This section contains detailed instructions to configure WebSphere Service Registry and Repository on z/OS. For a high level view of this information, see 4.4, “WebSphere Service Registry and Repository on z/OS” on page 110.

Configuring security

Exchange certificates with the WebSphere Message Broker for the security setup. To do this, extract the personal certificate setup for WebSphere Service Registry and Repository and save it in binary format so that it can be imported into the WebSphere Message Broker keystore.

Extracting the certificate

To extract the certificate in WebSphere Application Server:

1. Go to the **Security** section in the administrative console.
2. Select **SSL certificate and key management**.

3. Select the default certificate and click **Extract**. Figure B-6 shows the default certificate.

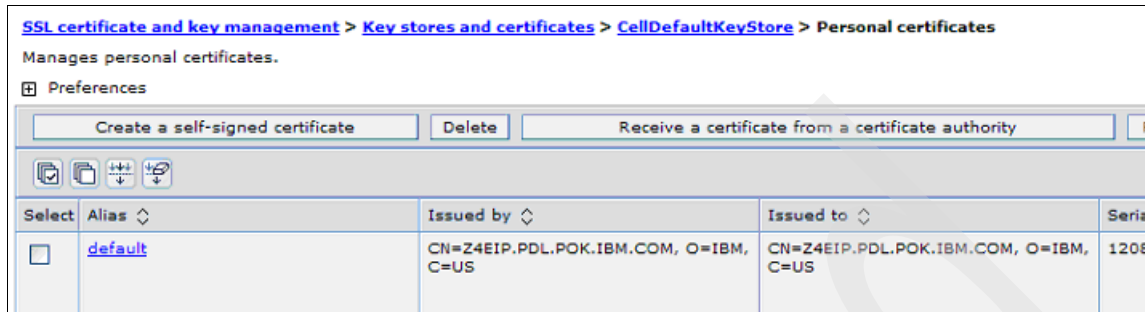


Figure B-6 Default certificate in SSL certificate and key management

4. In General Properties (Figure B-7), specify a file name and binary data type.

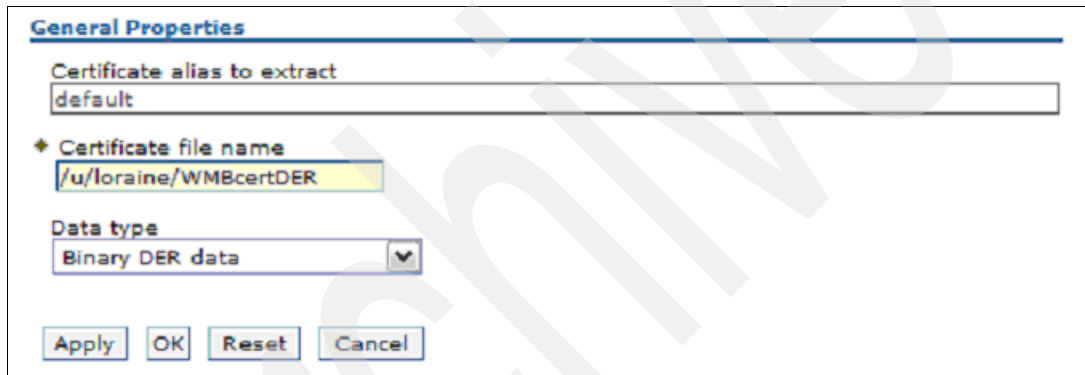


Figure B-7 The General Properties page

Keytool can be used to import the certificate into a truststore used by WebSphere Message Broker. For further explanation, see “Configuring a broker to access a secure WebSphere Service Registry and Repository server” on page 123.

SOA messaging WSDL

The WSDL in Example B-1 contains two (WebSphere MQ) endpoints and is used by the accept quote message flow. Port types and WebSphere MQ Internationalized Resource Identifiers (IRIs) are shown in bold.

Example: B-1 SOA messaging WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="AcceptService" targetNamespace="http://example.org"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:yc="http://example.org"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <wsdl:message name="acceptRequestMessage">
  </wsdl:message>
  <wsdl:portType name="AcceptLGIPortType">
    <wsdl:operation name="accept">
      <wsdl:input message="yc:acceptRequestMessage"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="AcceptLGIBinding" type="yc:AcceptLGIPortType">
    <soap:binding style="document"
      transport="http://www.ibm.com/xmlns/prod/wmq/transport"/>
    <wsdl:operation name="accept">
      <soap:operation soapAction="http://example.org/accept"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="AcceptLGI">
    <wsdl:port binding="yc:AcceptLGIBinding" name="AcceptLGIPort">
      <soap:address location="wmq:/msg/queue/LGI.ACCEPT.REQUEST"/>
    </wsdl:port>
  </wsdl:service>
  <wsdl:portType name="AcceptDCPortType">
    <wsdl:operation name="accept">
      <wsdl:input message="yc:acceptRequestMessage"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="AcceptDCBinding" type="yc:AcceptDCPortType">
    <soap:binding style="document"
      transport="http://www.ibm.com/xmlns/prod/wmq/transport"/>
    <wsdl:operation name="accept">
      <soap:operation soapAction="http://example.org/accept"/>
```

```
<wsdl:input>
  <soap:body use="literal"/>
</wsdl:input>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="AcceptDC">
  <wsdl:port binding="yc:AcceptDCBinding" name="AcceptDCPort">
    <soap:address location="wmq:/msg/queue/ACCEPT.REQUEST"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

Instructions for 4.5, 'WebSphere MQ on z/OS'

This section contains detailed instructions to configure WebSphere MQ on z/OS. For a high level view, see 4.5, "WebSphere MQ on z/OS" on page 113.

Configuring security

This section explains how to configure WebSphere MQ access control on z/OS to meet the requirements of the LGI scenario.

Configuring access control

The following task illustrates how to provide sufficient access to the message brokers and CICS regions to the queue manager resources they need for the LGI Web application. Note the following details for this task:

- ▶ The MQ Administrators ID is part of WebSphere MQADMIN RACF group.
- ▶ The system tasks IDs for the queue managers and channel initiators belong to the RACF group WebSphere MQTASKS.
- ▶ The started task user ID for the message broker is SYSTASK1.
- ▶ The local (z/OS) queue manager name is CSQ3, which is a member of the queue-sharing group MQGT. This is the same as the queue managers that are local to a WebSphere Application Server server described in Chapter 3, "Front-end deployment (AIX, Linux for System z, and z/OS)" on page 51.
- ▶ The local queue that is accessed by the message brokers to obtain the request messages is BROKER.QUOTE.REQUEST.

To configure access control:

1. Log on to a TSO session with a user ID that is part of the WebSphere MQADMIN RACF group.

2. Define the RACF profiles at the queue sharing group level and grant update access to all local request queues for the group. This action allows the brokers to obtain messages. Enter the following RACF commands:

```
RDEF MQQUEUE MQGT.BROKER.QUOTE.REQUEST UACC(NONE) OWNER(WebSphere
MQADMIN)
PE MQGT.BROKER.QUOTE.REQUEST CL(MQQUEUE) ID(WebSphere MQADMIN)
ACC(UPDATE)
PE MQGT.BROKER.QUOTE.REQUEST CL(MQQUEUE) ID(WebSphere MQTASKS)
ACC(UPDATE)
PE MQGT.BROKER.QUOTE.REQUEST CL(MQQUEUE) ID(MQSTEST) ACC(UPDATE)
SETR GENERIC(MQQUEUE) RACLIST(MQQUEUE) REFRESH
```

Note the following considerations for this step:

- All other queues that are only used by the message broker message flows use the same permissions as indicated in this step.
- If the request queue is shared and accessible by the queue managers local to the WebSphere Application Server server, you also need the permission specified in the following command:

```
PE MQGT.BROKER.QUOTE.REQUEST CL(MQQUEUE) ID(PSA2SRG) ACC(UPDATE)
```

3. Define the RACF profiles and grant the permissions required to the CICS regions for the WebSphere MQ-CICS Bridge initiation queue and request:

```
RDEFINE MQQUEUE MQGT.CICS*.** UACC(NONE)
PE MQGT.CICS*.** CL(MQQUEUE) ID(WebSphere MQADMIN) ACC(UPDATE)
PE MQGT.CICS*.** CL(MQQUEUE) ID(WebSphere MQTASKS) ACC(UPDATE)
PE MQGT.CICS*.** CL(MQQUEUE) ID(CICSUSER) ACC(UPDATE)
PE MQGT.CICS*.** CL(MQQUEUE) ID(CICSGRP) ACC(UPDATE)
```

4. Grant permission for the reply queue specified by the message broker:

```
PE MQGT.BROKER.QUOTE.REPLY CL(MQQUEUE) ID(CICSUSER) ACC(UPDATE)
PE MQGT.BROKER.QUOTE.REPLY CL(MQQUEUE) ID(CICSGRP) ACC(UPDATE)
```

5. Refresh the queue manager's security configuration to implement the following changes:

- a. Enter the following MQM command:

```
REFRESH SECURITY TYPE(MQQUEUE) CMDSCOPE(*)
```

- b. Enter the following MQM command:

```
RVERIFY SECURITY(MQSTEST)
```

6. Log off the MQ Administrator.

Configuring a z/OS queue manager to use SSL

The following task explains how to enable a z/OS queue manager to use SSL and to create a self-signed certificate for the queue manager. Note the following details for this task:

- ▶ The channel initiator address space ID is SYSTASK.
- ▶ The local (z/OS) queue manager name is ST03.
- ▶ The z/OS queue manager is WebSphere MQST03KEYRING.

Keep in mind the following considerations:

- ▶ This scenario does not use the ICSF to store certificates. Extra setup steps are required if you want to use ICSF.
- ▶ In the following task, RACF commands are issued from the TSO command interface, while logged on as a user with sufficient RACF authority to create key rings and certificates. Also, the MQM commands are issued from the ISPF MQ utilities interface.

To enable a z/OS queue manager to use SSL and create a key ring and self-signed certificate for the queue manager:

1. Grant the channel initiator address space ID access to RACF key rings by entering the following command:

```
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(SYSTASK)
ACCESS(UPDATE)
```

2. Enable the queue managers to use SSL by entering the following MQM command. The value must be greater than 1 but less than 50 to prevent storage allocation problems.

```
ALTER QMGR SSLTASKS(2)
```

3. Restart the channel initiator address space (CHINIT) to implement the change:

- a. Stop the channel initiator:

```
/ST03 STOP CHINIT
```

- b. Start the channel initiator:

```
/ST03 START CHINIT
```

4. Create the key ring for the queue manager:

```
RACDCERT ID(SYSTASK) ADDRING(WebSphere MQST03KEYRING)
```

5. Add the key ring to the queue manager by entering the following MQM command, which sets the queue manager's property SSL key repository (SSLKEYR):

```
ALTER QMGR SSLKEYR(WebSphere MQST03KEYRING)
```

6. Create a self-signed certificate for the local queue manager by entering the following RACF command:

```
RACDCERT ID(SYSTASK) GENCERT SUBJECTSDN(CN('ST03') O('LGI') C('GB'))
WITHLABEL('ibmWebSphereMQST03')
```
7. Add the self-signed certificate to queue manager's key ring by entering the following RACF command:

```
RACDCERT ID(SYSTASK) CONNECT(ID(SYSTASK) LABEL('ibmWebSphereMQST03')
RING(WebSphere MQST03KEYRING) USAGE(PERSONAL))
```
8. Export the local queue manager's public certificate (so that it is available to be added later into remote queue managers' key rings and key databases) by entering the following RACF command:

```
RACDCERT ID(SYSTASK) EXPORT(LABEL('ibmWebSphereMQST03'))
DSN('LGI.SSLKEY.ST03') FORMAT(CERTB64)
```
9. Refresh the queue manager's SSL settings for the changes to take effect, by entering the following MQM command:

```
REFRESH SECURITY TYPE(SSL)
```

Exchanging certificates with an AIX or Linux on System z queue manager

The following task shows the steps on a z/OS queue manager to enable SSL on sender and receiver cluster channels to and from a remote queue manager. For the contents of this section, the remote queue managers can be either on AIX or Linux for System z. The behavior is the same unless otherwise specified. Note the following details for this task:

- ▶ The channel initiator address space ID is SYSTASK.
 - ▶ The local (z/OS) queue manager name is ST03.
 - ▶ The z/OS queue manager is WebSphere MQST03KEYRING.
 - ▶ The remote queue manager name is LGI.WAS.Z.01.
1. In the following task, we issued RACF commands from the ISPF TSO command interface while logged on as a user with sufficient RACF authority to create key rings and certificates. We issued the MQM commands from the ISPF MQ utilities interface.

To enable SSL on a z/OS queue manager's cluster sender and receiver channels to a remote queue manager:

1. Configure the z/OS queue manager to use SSL by following the steps in "Configuring a z/OS queue manager to use SSL" on page 215.
2. Obtain the public certificate for the remote queue manager, for example, LGI.WAS.Z.01. The queue manager's certificate is extracted to the remote system's temporary directory as a Base64-encoded ASCII data certificate.

3. Use the ISPF dataset utilities to create a z/OS sequential file into which remote public certificate will be copied (for example, 'CPIT.SSLKEY.AIXMQ') with the following format:
 - Organization = PS
 - Record Format = VB
 - Record Len = 84
 - Block Size = 27998
 - First and second extent tracks = 5

Error message on AIX: Our attempts to add a certificate to a key ring from a sequential file created dynamically during an FTP from AIX resulted in the error message: “An error occurred with the specified input dataset.”

4. Send the Base64-encoded ASCII data certificate file in ASCII mode by using FTP from the temporary directory to the precreated z/OS sequential file.
5. Add the remote queue manager's public certificate to local queue manager's RACF database:

```
RACDCERT ID(SYSTASK) ADD('CPIT.SSLKEY.AIXMQ') TRUST  
WITHLABEL('ibmWebSpheremqlgi.was.z.01')
```

Note the following considerations:

- For AIX, when adding certificates from AIX queue managers RACF displays, we encountered the informational message: “Certificate Authority not defined to RACF. Certificate added with TRUST status.”
 - For Linux for System z, when adding certificates from Linux on System z, we encountered the informational message: “The new profile for DIGTCERT will not be in effect until a SETROPTS REFRESH has been issued. The certificate that you are adding is self-signed. The certificate is added with TRUST status.”
 - If all z/OS queue managers share the same RACF database in a Parallel Sysplex environment, only enter this command once.
6. Connect the remote queue manager's public certificate to the local queue manager's key ring:

```
RACDCERT ID(SYSTASK) CONNECT(ID(SYSTASK)  
LABEL('ibmWebSpheremqlgi.was.z.01') RING(WebSphere MQST03KEYRING)  
USAGE(PERSONAL))
```

Repeat this command for each of the local z/OS queue managers.

7. Enter the following RACF commands to apply the changes:

```
SETROPTS RACLIST (DIGTCERT) REFRESH  
SETROPTS RACLIST (DIGTRING) REFRESH
```

8. For the changes to take effect, refresh the queue manager's SSL settings by entering the MQM command:

```
REFRESH SECURITY TYPE(SSL)
```

9. Set the cipher specification on the sender channel to match the value specified on the corresponding receiver channel on the remote queue manager by entering the following MQM command:

```
ALTER CHANNEL('senderChannelName') CHLTYPE(CLUSSDR)  
SSLCIPH(RC4_MD5_US)
```

10. Set the cipher specification to be used on the receiver channel and request client authentication by entering the following MQM command:

```
ALTER CHANNEL('receiverChannelName') CHLTYPE(CLUSRCVR)  
SSLCIPH(RC4_MD5_US) SSLCAUTH(REQUIRED)
```

11. Restart both ends of each cluster sender and receiver channel for the SSL settings to take effect. This includes the dynamic cluster sender channels. The dynamic cluster sender channels automatically inherit the same SSLCIPH value as the corresponding receiver channel after they are restarted.

Exchanging certificates with a WebSphere Process Server MQ link

The following task shows the steps that are required on a z/OS queue manager to enable SSL on sender and receiver channels to a service integration bus on WebSphere Process Server by using an MQ link. These steps apply to a service integration bus in any WebSphere Application Server v6.1-based product. Note the following details for this task:

- ▶ The channel initiator address space ID is SYSTASK.
- ▶ The local (z/OS) queue manager name is ST03.
- ▶ The z/OS queue manager is WebSphere MQST03KEYRING.
- ▶ The name of the certificate extracted from WebSphere Process Server for the MQ link is BPCBusKey.der.

We issued RACF commands from the ISPF TSO command interface while logged on as a user with sufficient RACF authority to create key rings and certificates. We issued MQM commands from the ISPF MQ utilities interface.

To enable SSL on a z/OS queue manager's sender and receiver channels to MQ link channels on a WebSphere Process Server service integration bus:

1. Configure the z/OS queue manager to use SSL by following the steps in "Configuring a z/OS queue manager to use SSL" on page 215.
2. Obtain the public certificate for the remote WebSphere Process Server MQ link. For example, the BPCBusKey.der certificate is extracted to a z/OS UNIX System Services temporary directory as binary DER data.

3. Use ISPF dataset utilities to create a z/OS sequential file into which remote public certificate will be copied (for example, 'CPIT.SSLKEY.BPCBUS') with the following format:
 - Organization = PS
 - Record Format = VB
 - Record Len = 84
 - Block Size = 27998
 - First and second extent tracks = 5

Error message with FTP: Our attempts to add a certificate to a key ring from a sequential file, created dynamically during FTP, resulted in the error message: “An error occurred with the specified input dataset.”

4. Send the file in binary mode by using FTP from the z/OS UNIX System Services temporary directory to precreated sequential file.
5. Add the remote MQ link (service integration bus) public certificate to the local queue manager's key ring by entering the following RACF commands:

```
RACDCERT ID(SYSTASK) ADD('CPIT.SSLKEY.BPCBUS ') TRUST
WITHLABEL('BPCBusKey')
RACDCERT ID(SYSTASK) CONNECT(ID(SYSTASK) LABEL('BPCBusKey')
RING(WebSphere MQST03KEYRING) USAGE(PERSONAL))
```
6. For the changes to take effect, refresh the queue manager's SSL settings by entering the following MQM command:

```
REFRESH SECURITY TYPE(SSL).
```
7. Set the cipher specification on the sender channel to match the value that is specified on the corresponding MQ link receiver channel by entering the following MQM command:

```
ALTER CHANNEL('senderChannelName') CHLTYPE(SDR) SSLCIPH(RC4_SHA_US)
```
8. Set the cipher specification to be used on the receiver channel and request client authentication:

```
ALTER CHANNEL('receiverChannelName') CHLTYPE(RCVR)
SSLCIPH(RC4_SHA_US) SSLCAUTH(REQUIRED)
```
9. Restart both ends of each sender and receiver channel for the SSL settings to take effect.

Useful RACDCERT commands

In addition to the commands documented in the previous sections, the following commands can be useful:

- ▶ List personal certificates defined to a user:
`RACDCERT ID(user) LIST`
- ▶ List personal certificates defined to user in a specific key ring:
`RACDCERT ID(user) LISTRING(keyring)`
- ▶ Remove a certificate from a key ring:
`RACDCERT ID(user) REMOVE(LABEL('label') RING(keyring))`
- ▶ Delete a specified personal certificate:
`RACDCERT ID(user) DELETE(LABEL('label'))`
- ▶ Delete a specified key ring associated with a user:
`RACDCERT ID(user) DELRING(keyring)`

Troubleshooting WebSphere MQ SSL configuration errors

The following common runtime problems can occur after configuring WebSphere MQ SSL. We include suggestions for possible solutions.

- ▶ **Problem:** Your attempt to start the channel from the WebSphere MQ queue manager to the MQ link on WebSphere Process Server (that you defined with client authentication) fails. You see the following messages in the WebSphere MQ Channel Initiator log:

```
CSQX500I ST03 CSQXRCTL Channel channelName started
```

```
CSQX685E ST03 CSQXRCTL No self-signed SSL certificate for channel  
channelName
```

```
CSQX599E ST03 CSQXRCTL Channel channelName ended abnormally
```

Solution: Add the WebSphere Process Server MQ link certificate to the WebSphere MQ queue manager's key ring.

- ▶ **Problem:** Your attempt to start the channel from the MQ link on WebSphere Process Server to the WebSphere MQ queue manager (that you defined with client authentication) fails. You see the following messages in the WebSphere MQ Channel Initiator log:

```
CSQX685E ST03 CSQXRCTL No self-signed SSL certificate for channel  
channelNameF
```

You also see the following messages in the WebSphere Process Server Adjunct region's SYSPRINT log:

BB000220E: CWSIQ0017E: The sender channel *channelName* for MQLink *MQLinkName* has failed to establish a connection with the remote host *QMgrHostname* because the remote listener for port *QMgrPort* is not available.

Solution: Add the WebSphere Process Server MQ link certificate to the WebSphere MQ queue manager's key ring.

- **Problem:** Your attempt to start the channel from the WebSphere MQ queue manager to the MQ link on WebSphere Process Server (that you defined with client authentication) fails. You see the following messages in the WebSphere MQ Channel Initiator log:

CSQX634E ST03 CSQXRCTL SSL certificate failed remote check, channel *channelName*, connection *QMgrHostname*

Solution: Add the WebSphere MQ queue manager's certificate to the WebSphere Process Server MQ link key database.

- **Problem:** Your attempt to start the channel from the MQ link on WebSphere Process Server to the WebSphere MQ queue manager (that you defined with client authentication) fails. You see the following messages in the WebSphere Process Server Adjunct region's SYSPRINT log:

BB000222I: CWSIC3201I: The sending side of the WebSphere MQ Link *MQLinkName* has been requested to start.

CWPKI0022E: SSL HANDSHAKE FAILURE: A signer with SubjectDN "CN=ST03, O=LGI, C=GB" was sent from target host:port "*:5579". The signer may need to be added to local truststore "*WAS_HOME*/profiles/*profileName* /config/cells/*cellName*/*truststoreName*.p12" located in SSL configuration alias "*SSLConfigName*" loaded from SSL configuration file "security.xml". The extended error message from the SSL handshake exception is: "PKIX path building failed: java.security.cert.CertPathBuilderException: unable to find valid certification path to requested target".

Solution: Add the WebSphere MQ queue manager's certificate to the WebSphere Process Server MQ link key database.

Instructions for 4.6, ‘WebSphere Message Broker on z/OS’

This section contains detailed instructions to configure WebSphere Message Broker. For a high level view of this information, see 4.6, “WebSphere Message Broker on z/OS” on page 122.

Configuring security

This section provides detailed information about the steps that are required to configure WebSphere Message Broker to access a secured WebSphere Service Registry and Repository server.

Configuring a message broker to access a secure WebSphere Service Registry and Repository server

The following task shows how to allow a message broker to access a secure WebSphere Service Registry and Repository server. This task is based on the instructions in “Accessing a secure WebSphere Service Registry and Repository” in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac56150_.htm

Note the following details in this task:

- ▶ The certificate extracted from WebSphere Service Registry and Repository (in Binary DER format) is `WSRRCertificate.der`.
- ▶ The host name of the WebSphere Service Registry and Repository server is `WSRRHostname`.
- ▶ The `WC_defaulthost_secure` port of the WebSphere Service Registry and Repository server is `WSRRPort`.
- ▶ The user ID authorized to perform queries in WebSphere Service Registry and Repository is `WSRRUser`.
- ▶ The password for user ID `WSRRUser` is `WSRRPassword`.
- ▶ The name and location for the broker's keystore is `/brokers/Brokerkeystore.jks`.
- ▶ The name and location for the broker's truststore is `/brokers/Brokertruststore.jks`.
- ▶ The name of the message broker is `ST03BRK`.

To configure a message broker to access a secure WebSphere Service Registry and Repository server:

1. Obtain the WebSphere Service Registry and Repository server's HTTP public certificate. We assume that administration and application security are enabled on the WebSphere Service Registry and Repository cluster and custom certificates have been created for use by connections using its WC_defaulthost_secure port.
 - a. By using the WebSphere Service Registry and Repository Administration Console to extract the public certificate, select **Security**.
 - b. Select **SSL certificate and key management**.
 - c. Select **Key stores and certificates**.
 - d. Select the name of the keystore that holds the WebSphere Service Registry and Repository server's personal certificate (used for connections on its WC_defaulthost_secure port).
 - e. Select **Personal Certificates**.
 - f. Select the required certificate.
 - g. Click **Extract self-signed certificate**.
 - h. Select the option to extract the certificate as **Binary DER**, and specify a name (for example, WSRRCertificate.der) and local directory (accessible to the Administration Console user) into which the public certificate will be extracted.
2. Create a keystore for the message broker.

Key management tool: During our test activities, we were unable to find a key management tool in the WebSphere Message Broker directories on z/OS. Therefore, we used the **ikeyman** utility that ships with WebSphere Application Server 6.1 on z/OS rather than the key management tool that is available with the Message Broker Toolkit on Windows. In addition, to use the GUI version of the tool, ensure that DISPLAY is exported to a suitable machine with X11 support.

- a. Launch the key management tool from a UNIX System Service terminal:
WAS_HOME/profiles/profileName/bin/ikeyman.sh.
- b. Select **Key Database file** and then click **New**.
- c. For Key database type, select **JKS**.
- d. For File Name, type Brokerkeystore.jks.
- e. For Location, type /brokers.
- f. Click **OK**.

- g. For Password, type a value.
- h. Close the key database.
3. Create a truststore for the message broker by repeating the previous step. This time, for File Name, type `Broketruststore.jks`.
4. Import the WebSphere Service Registry and Repository certificate into the broker's truststore:
 - a. Open the broker's truststore, following the previous instructions if needed.
 - b. Select **Signer Certificates** from the list.
 - c. Click **Add**.
 - d. For Data Type, select **Binary DER data**.
 - e. Navigate to the certificate extracted from WebSphere Service Registry and Repository (for example, `WSRRCertificate.der`) and specify an arbitrary value for the label (for example, `WSRRCertificate`).
 - f. Close the truststore and exit from `ikeyman`.
5. Ensure that the message broker address space user IDs have read access to the keystore and truststore.
6. Modify the broker's configuration to set the URL of the WebSphere Service Registry and Repository server's HTTPS port by editing '`COMPONENTDATASET(BIPCHPR)`' to issue the `mqsichangeproperties` command:


```
mqsichangeproperties ST03BRK -c ServiceRegistries -o DefaultWSRR -n
endpointAddress -v https://WSRRHostname:
WSRRPort/WSRRCoreSDO/services/WSRRCoreSDOPort
```
7. Modify the broker's configuration to access its keystore and truststore by editing '`COMPONENTDATASET(BIPCHPR)`' to issue the `mqsichangeproperties` command:


```
mqsichangeproperties ST03BRK -o BrokerRegistry -n brokerKeystoreFile
-v /brokers/Brokerkeystore.jks
mqsichangeproperties ST03BRK -o BrokerRegistry -n
brokerTruststoreFile -v /brokers/Brokertruststore.jks
```
8. Stop the broker. The broker must be stopped to run the `mqsisetdbparms` commands. The broker must be restarted before changes made through the `mqsichangeproperties` command can take effect.
9. Set the user ID and password used by the broker to access WebSphere Service Registry and Repository server by editing '`COMPONENTDATASET(BIPSDBP)`' to issue the `mqsisetdbparms` command:


```
mqsisetdbparms ST03BRK -n DefaultWSRR::WSRR -u WSRRUser -p
WSRRPassword
```


10. Set the user ID and password that is used by the broker to access its keystore and truststore by editing '*COMPONENTDATASET(BIPSDBP)*' to issue the **mqsisetdbparms** command:

```
mqsisetdbparms ST03BRK -n brokerKeystore::password -u dummy -p  
keystore-password  
mqsisetdbparms ST03BRK -n brokerTruststore::password -u dummy -p  
truststore-password
```

The user ID value is not used to access the brokerKeystore or brokerTruststore. However, the command syntax requires a value. Therefore, for the purposes of our test activities, we used *dummy*.

11. Configure the user ID and password that is used by the broker to access the jms resources in WebSphere Service Registry and Repository used by Cache Notification.

If the broker's configuration property `enableCacheNotification` is set to true, you must complete the task as documented in the "Setting up Cache Notification" topic in the WebSphere Message Broker 6.1 Information Center at the following address:

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac56080_.htm

Set the user ID and password that are required to access the WebSphere Service Registry and Repository server's JMS connection factory by issuing the following command:

```
mqsisetdbparms ST03BRK -n jms::DefaultWSRR@jms/SRConnectionFactory  
-u WSRRUser -p WSRRPassword
```

12. Start the broker.

13. Confirm the broker's property values by editing '*COMPONENTDATASET(BIPRPPR)*' to issue the **mqsireportproperties** command:

```
mqsireportproperties ST03BRK -o BrokerRegistry -a  
mqsireportproperties ST03BRK -o DefaultWSRR -c ServiceRegistries -r
```

Instructions for 4.9, 'WebSphere MQ File Transfer Edition on AIX'

This section contains detailed instructions for configuring WebSphere MQ FTE on AIX. For a high level view of this information, see 4.9, "WebSphere MQ File Transfer Edition on AIX" on page 130.

Configuring the AIX coordination queue manager

To configure a WebSphere MQ FTE agent:

1. Connect the coordination queue manager to the agent queue managers. This involves connecting the coordination queue manager to the cluster and defining the appropriate SSL artifacts. See 3.5, “WebSphere MQ on AIX and Linux for System z” on page 70, for details about connecting the cluster queue managers with SSL channels.
2. Either run the `LGI.BACK.FTE.CQM.mqsc` against the coordination queue manager or define the objects manually. The MQSC file is in the data directory of a configured agent (for example `/var/fte/LGI.BACK.FTE.CQM/LGI.BACK.FTE.CQM.mqsc`). The following object definitions are in the file and can be used to manually define the required objects:

```
DEFINE TOPIC('SYSTEM.FTE') TOPICSTR('SYSTEM.FTE')
  REPLACE
DEFINE QLOCAL(SYSTEM.FTE)
  LIKE(SYSTEM.BROKER.DEFAULT.STREAM) REPLACE
ALTER NAMELIST(SYSTEM.QPUBSUB.QUEUE.NAMELIST)
  NAMES(SYSTEM.BROKER.DEFAULT.STREAM,
        SYSTEM.BROKER.ADMIN.STREAM,SYSTEM.FTE)
```

3. Define WebSphere MQ security users (and an appropriate group) for agent users so that each agent's user is defined on the coordination queue manager system.
4. Configure WebSphere MQ security (using `setmqaut`) to give publish access to the agent users.

Checking the configuration: After the coordination queue manager and at least one agent is configured, check the configuration by using the `fteListAgents` command.

Instructions for 4.10, ‘WebSphere MQ File Transfer Edition on z/OS’

This section explains how to configure WebSphere MQ FTE on z/OS. For a high level view of this information, see 4.10, “WebSphere MQ File Transfer Edition on z/OS” on page 132.

Note: WebSphere MQ FTE runs in UNIX System Services.

Configuring the z/OS agent

To configure a WebSphere MQ FTE agent:

1. Install WebSphere MQ FTE on UNIX System Services. Choose the server installation option. Only bindings connections are supported from a UNIX System Services environment to a z/OS queue manager.
2. Ensure that WebSphere MQ fix pack 6.0.2.5 is installed on the z/OS queue manager.
3. Before creating a WebSphere MQ FTE agent, ensure that a coordination queue manager is available. See 4.9, “WebSphere MQ File Transfer Edition on AIX” on page 130, for installing and configuring the coordination queue manager.
4. Optional: Set the PATH environment variable to include the WebSphere MQ FTE bin directory:

```
export PATH=$PATH:/usr/fte/bin
```
5. Set the FTE_JAVA_HOME environment variable to the Java 5 installation path:

```
export FTE_JAVA_HOME=/usr/java5
```
6. Set the FTE_CONFIG environment variable to the WebSphere MQ FTE data directory. This directory contains all configuration and status information for the WebSphere MQ FTE installation.

```
export FTE_CONFIG=/var/fte
```
7. Set the STEPLIB environment variable to include the WebSphere MQ SCSQAUTH, SCSQLOAD, and SCSQANLE data sets:

```
export  
STEPLIB=$STEPLIB:MQM.V600.SCSQAUTH:MQM.V600.SCSQANLE:MQM.V600.SCSQLOAD
```
8. Create the files that are used to connect to the coordination queue manager. The following command creates the data directory (for example, /var/fte), the coordination queue manager subdirectory (for example, /var/fte/LGI.BACK.FTE.CQM), the wmqfte.properties file, and the coordination.properties file:

```
fteSetupCoordination -coordinationQMGr LGI.BACK.FTE.CQM
```
9. Create files that are used to connect to the command queue manager. In this case, the command queue manager is the local queue manager to which the agent will connect. This creates the command.properties file:

```
fteSetupCommands -connectionQMGr BE01
```

10. Create an agent, which creates an agent subdirectory (for example, `/var/fte/LGI.BACK.FTE.CQM/agents/BE01`) that includes the `BE01_create.mqsc` file:

```
fteCreateAgent -agentName BE01 -agentQMGr BE01
```
11. Create the queues that are required by the agent. Using the commands in the `BE01_create.mqsc` file as a guide, use MQSC (for example, in TSO) or a WebSphere MQ administration tool (for example, the WebSphere MQ Explorer).
12. Start the agent as a background process:

```
fteStartAgent -F BE01
```
13. Check that the agent is running:

```
fteListAgents
```
14. If the agent is not listed, check the agent log for errors and check for FFDCs. Both the log file and any FFDCs are in agent logs directory (for example, `/var/fte/LGI.BACK.FTE.CQM/agents/BE01/logs`). Also check the status of the WebSphere MQ channels between the agent queue manager, the coordination queue manager, and the WebSphere MQ error logs.

Stopping agents: In addition to starting and listing agents, agents can also be stopped by using the `fteStopAgent` command.

Instructions for 4.11, 'IBM Tivoli Monitoring on AIX'

This section explains how to configure IBM Tivoli Monitoring on AIX. For a high level view of this information, see 4.11, "IBM Tivoli Monitoring on AIX" on page 134. The configuration requires the installation of DB2, which is used for data warehousing.

Installing DB2 Enterprise Server Edition V8.1

To install DB2 Enterprise Server Edition, V8.1 (also known as 8.2.9) for AIX (32-bit):

1. Log in as root and enter the following command:

```
ulimit -f
```

If the result is not *unlimited*, modify the following line in the `/etc/security/limits` file:

```
fsize= -1
```

For changes to the `/etc/security/limits` file to take effect, log out and then log in again.

2. Start the installation wizard as explains in the DB2 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v8//topic/com.ibm.db2.udb.doc/start/t0008875.htm>

3. Choose custom installation and select the following components:

- Server support
- Client support
- Administration tools
- Application development tools
- Getting Started
- First steps
- Sample database support

4. When prompted, choose the following options:

- Application development Select Create a DB2 instance - 32 bit
- Single-partition instance
- Do not prepare the DB2 tools catalog on this computer
- Local - Create a contact list on this system

Installing Fix Pack 16

To install Fix Pack 16 for DB2 Enterprise Server Edition, V8.1 (also known as 8.2.9) for AIX (32-bit):

1. Stop all DB2 processes. Switch to `db2inst1` authority (**su - db2inst1**) and enter the following commands:

```
db2 terminate
db2stop
db2licd -end
exit
```

2. Disable and stop the Fault Monitor. Switch to root authority (**su - root**) and enter the following commands:

```
/usr/opt/db2_08_01/bin/db2fmcu -d
/usr/opt/db2_08_01/bin/db2fm -i dasusr1 -D
```

3. Stop the DB2 Admin server. Switch to `db2usr1` authority (**su - dasusr1**) and enter the following commands:

```
db2admin stop
exit
```

4. Ensure that all DB2 interprocess communications are cleaned for the instance to be updated. Switch to db2inst1 authority (**su - db2inst1**) and enter the following commands:

```
/home/db2inst1/sqllib/bin/ipclean  
exit
```

5. Switch to root authority (**su - root**).
6. Remove any currently unused modules in kernel and library memory:

```
/usr/sbin/slibclean
```

7. Install the fix pack:

```
installFixPak -y
```

8. Update instances to use the new level of DB2:

```
/usr/opt/db2_08_01/instance/db2iupdt db2inst1  
/usr/opt/db2_08_01/instance/dasupdt dasusr1  
exit
```

9. Restart DB2. Switch to db2inst1 authority (**su - db2inst1**) and run the following commands:

```
db2start  
db2level
```

10. Verify the DB2 installation by following the steps in the “Verifying the installation using the command line processor (CLP)” topic in the DB2 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/db2luw/v8//topic/com.ibm.db2.udb.doc/start/t0006839.htm>

Configuring IBM Tivoli Monitoring

The following sections explain how to configure IBM Tivoli Monitoring.

Installing IBM Tivoli Monitoring

To install IBM Tivoli Monitoring:

1. Set the File size and Data limit to unlimited:
 - a. Log in as root.
 - b. Modify the following lines in the `/etc/security/limits` file:

```
data = -1  
fsize = -1
```
 - c. Log out and then log in again for changes to take effect.

2. Start the IBM Tivoli Monitoring install wizard with the following command:
`install.sh`
3. Choose the following options when prompted:
 - Product package AIX R5.3 (64 bit) and all components (for example, Tivoli Enterprise Monitoring Server)
 - Tivoli Enterprise Portal Browser Client support and all components (for example, Monitoring Agent for UNIX Logs)
 - Tivoli Enterprise Portal Server support and all components (for example, Monitoring Agent for UNIX Logs)

Configuring Tivoli Enterprise Portal Server

Configure Tivoli Enterprise Portal Server:

1. Type the following command:

```
/opt/IBM/ITM/bin/itmcmd config -A cq
```

2. Enter the following information (in bold) when prompted:

Will this agent connect to a CMS? [YES or NO] (Default is: YES):

Response: **Enter**

CMS Host Name (Default is: p6b2lp04):

Response: **Enter**

Network Protocol [ip, sna, ip.pipe, or ip.spipe] (Default is: ip.pipe):

Response: **ip.pipe**

Now choose the next protocol from one of these: - ip - sna - ip.spipe - none Network Protocol 2 (Default is: none):

Response: **ip.spipe**

Now choose the next protocol from one of these: - ip - sna - none Network Protocol 3 (Default is: none):

Response: **Enter**

IP.PIPE Port Number (Default is: 1918):

Response: **Enter**

Enter name of KDC_PARTITION (Default is: null): Response: **Enter**

IP.SPIPE Port Number (Default is: 3660):

Response: **Enter**

Enter Optional Primary Network Name or "none" (Default is: none):

Response: **Enter**

Enable SSL for TEP Clients (y/n) (Default is: N): Response: **Enter**

Enter the DB2 instance name (Default is: db2inst1): Response: **Enter**

Enter the DB2 admin ID (Default is: db2inst1): Response: **Enter**

Enter the DB2 admin ID (Default is: db2inst1): Response: *********

Re-type: Enter the password for the DB2 admin ID: Response: *********

Enter the TEPS DB2 database name (Default is: TEPS):

Response: **Enter**
Enter the TEPS DB2 database login ID(Default is: itmuser):
Response: **Enter**
Enter the password for the TEPS DB2 database login ID:
Response: *********
Re-type: Enter the password for the TEPS DB2 database login ID:
Response: *********
Is it OK to create the TEPS login ID if not found (y/n))(Default is: Y):
Response: **Enter**
Are you using DB2 or Oracle(JDBC) for Warehouse? [DB2 or JDBC] (Default is: DB2):
Response: **Enter**
Enter the Warehouse database name(Default is: WAREHOUS):
Response: **Enter**
Enter the Warehouse user ID(Default is: itmuser): Response: **Enter**
Enter the password for the Warehouse user ID: Response: *********
Re-type:Enter the password for the Warehouse user ID:
Response: *********

Creating the DB2 Data Warehouse database

To create the DB2 Data Warehouse database:

1. Switch to db2inst1 authority:
`su - db2inst1`
2. Type the following command:
`db2 create database warehous using codeset utf-8 territory US`
3. Configure the DB2 Data Warehouse database. Start by opening the DB2 Command Line Processor:
`db2`
4. Type the following DB2 commands:
`db2 => connect to warehous`
`db2 => create bufferpool ITMBUF8K immediate size 250 pagesize 8 k`
`db2 => create regular tablespace ITMREG8K pagesize 8 k managed by system using ('itmreg8k') bufferpool ITMBUF8K`
`db2 => create system temporary tablespace ITMSYS8K pagesize 8 k managed by system using ('itmsys8k') bufferpool ITMBUF8K`
`db2 => create user temporary tablespace ITMUSER8K pagesize 8 k managed by system using ('itmuser8k') bufferpool ITMBUF8K`


```
db2 => grant connect on database to user itmuser
db2 => grant createtab on database to user itmuser
db2 => quit
```

5. Type the following commands:

```
db2set -i db2inst1 DB2COMM=tcPIP
db2 update dbm cfg using SVCENAME 60000
db2stop
db2start
exit
```

Starting Tivoli Enterprise Management Server and Tivoli Enterprise Portal Server

To start Tivoli Enterprise Management Server and Tivoli Enterprise Portal Server:

1. Start Tivoli Enterprise Management Server:

```
/opt/IBM/ITM/bin/itmcmd server start HUB_BSATEMS
```

2. Start Tivoli Enterprise Portal Server:

```
/opt/IBM/ITM/bin/itmcmd agent start cq
```

Configuring the Warehouse Proxy agent

To configure the Warehouse Proxy agent:

1. Ensure that you have an X Window System running.
2. Start the Manage Tivoli Enterprise Monitoring Services window:

```
/opt/IBM/ITM/bin/itmcmd manage
```
3. Select **Warehouse Proxy**.
4. Right-click **Warehouse Proxy** and select **Configure**.
5. Click the **Agent Parameters** tab.
6. Add the names and directory locations of the JDBC driver Java archive (JAR) files to the JDBC Drivers list box:
 - a. Scroll the bar at the bottom to display the Add and Delete buttons, which are to the right of the JDBC drivers list box.
 - b. Click **Add** to display the file browser window.
 - c. Navigate to the location of the driver files (/usr/opt/db2_08_01/java).

- d. Select the following driver files:
 - Db2jcc.jar
 - Db2jcc_license_cu.jar
- e. Click **OK** to close the browser window and add the JDBC driver files to the list.
7. Change the default value displayed in the Warehouse URL field if it is not correct. The correct value is `dbc:db2://localhost:60000/WAREHOUSE`.
8. Verify the JDBC driver name in the Warehouse Driver field. The DB2 JDBC driver name is `com.ibm.db2.jcc_DB2Driver`.
9. Enter the Warehouse user (`itmuser`).
10. Enter the Warehouse password.
11. Click **Test database connection**.
12. Click **Save**.
13. From the Manage Tivoli Enterprise Monitoring Services window, start the Warehouse Proxy. Click **Warehouse Proxy** to highlight the service, right-click **Warehouse Proxy** and select **Start**.

Starting the Tivoli Enterprise Portal Server

To start the Tivoli Enterprise Portal Server:

1. Type the following command:
`opt/IBM/ITM/bin/itmcmd agent start cq`
2. Click **Tivoli Enterprise Portal Server** to highlight the service, right-click **Tivoli Enterprise Portal Server**, and select **Configure**.

Configuring the Summarization and Pruning agent

To configure the Summarization and Pruning agent:

1. Ensure that you have an X Window System running.
2. Start the Manage Tivoli Enterprise Monitoring Services window:
`/opt/IBM/ITM/bin/itmcmd manage`
3. Select **Summarization and Pruning** to highlight the service, right-click **Summarization and Pruning**, and select **Configure**.
4. Click the **Agent Parameters** tab.

5. Add the names and directory locations of the JDBC driver JAR files to the JDBC Drivers list box:
 - a. Scroll the bar at the bottom to display the Add and Delete buttons, which are to the right of the JDBC drivers list box.
 - b. Click **Add** to display the file browser window.
 - c. Navigate to the location of the driver files (/usr/opt/db2_08_01/java).
 - d. Select the following driver files:
 - Db2jcc.jar
 - Db2jcc_license_cu.jar
 - e. Click **OK** to close the browser window and add the JDBC driver files to the list.
6. Change the default value displayed in the Warehouse URL field if it is not correct. The correct value is dbc:db2://localhost:60000/WAREHOUSE.
7. Verify the JDBC driver name in the Warehouse Driver field. The DB2 JDBC driver name is com.ibm.db2.jcc.DB2Driver.
8. Enter the Warehouse user (itmuser).
9. Enter the Warehouse password.
10. Click **Test database connection**.
11. Click **Save**.
12. Start the Warehouse Proxy from the Manage Tivoli Enterprise Monitoring Services window. Select **Summarization and Pruning** to highlight the service, right-click **Summarization and Pruning**, and select **Start**.

Verifying the installation

To verify the installation:

1. Ensure that you have an X Window System running.
2. Start the Manage Tivoli Enterprise Monitoring Services window:
`/opt/IBM/ITM/bin/itmcmd manage`
3. Start the Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server.
4. Open Internet Explorer v5 or later and navigate to the following URL:
`http://<hostname>:1920//cnp/client`

Installing IBM Tivoli Monitoring application support for IBM Tivoli Composite Application Manager for WebSphere

To install IBM Tivoli Monitoring application support for IBM Tivoli Composite Application Manager for WebSphere:

1. Install Tivoli Enterprise Management Server application support by following the instructions in the “Installing agent support files” topic in the information center at the following address:

<http://publib.boulder.ibm.com/infocenter/imshep1/v3r0/topic/com.ibm.wrs61.doc/instagentsupfiles.html>

2. Install Tivoli Enterprise Portal Server application support by using the same instructions in step 1. However, be sure to use Tivoli Enterprise Portal Server, instead of Tivoli Enterprise Management Server, in all steps.

Instructions for 5.1, ‘Web Services Security on AIX and z/OS’

This section explains how to configure and deploy Web Services Security (WS-Security) in the LGI scenario. For a high level view of this information, see 5.1, “Web Services Security on AIX and z/OS” on page 136.

Preparation

To configure WS-Security 1.0 specification, the applications to be secured must be at a supported level. Follow this guidance:

- Ensure that WebSphere Application Server-based Web service applications are at J2EE level 1.4. This level is required to configure WS-Security at the 1.0 specification level.

Rational Application Developer: Rational Application Developer 7.0 can be used to migrate existing Web service application projects to the J2EE 1.4 specification.

- ▶ Ensure that the WebSphere Message Broker consumer or provider message flows implement the SOAP nodes that are introduced in V6.1. WebSphere Message Broker 6.1 provides support for WS-Security over SOAP/HTTP.

Note: WebSphere Message Broker Toolkit 6.1 provides wizards to develop Web service consumer or provider message flows by using these SOAP nodes.

Configuring WS-Security

The following references provide WS-Security configuration examples for WebSphere Application Server and WebSphere Message Broker environments. We used these references as a basis for securing the Web service calls in the LGI scenario:

- ▶ For information about the Rational Application Developer and WebSphere Integration Developer WS-Security configuration wizards, see the “Securing JAX-RPC Web services using Web services security wizards” topic in the Rational Application Developer 7.0 Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/radhelp/v7r0m0/index.jsp?topic=/com.ibm.etools.webservice.security.wizard.doc/topics/topwssec.html>

Tip: If you applied maintenance to your Rational Application Developer v7 installation and are unable to locate the “Web Services tab in the Project Explorer view” described in the information center instructions, expand the **JSR-109 Web Services** project in the Project Explorer view.

- ▶ A good overview and example of configuring WS-Security for WebSphere Application Server is in Chapters 8 and 25 of *Web Services Handbook for WebSphere Application Server 6.1*, SG24-7257:
- ▶ Another example of configuring WS-Security for WebSphere Application Server is in the series of developerWorks articles “IBM WebSphere Developer Technical Journal: Web services security with WebSphere Application Server V6” (in particular Parts 2 and 3) at the following address:

http://www.ibm.com/developerworks/websphere/techjournal/0603_cowan/0603_cowan.html

- ▶ A good example of configuring WebSphere Message Broker to sign Web service calls is available in the developerWorks article “Signing flows for Web Services Security” at the following address:

<http://www.ibm.com/developerworks/webservices/library/ws-security/>

The following references are helpful for WS-Security implementations in WebSphere Application Server and WebSphere Message Broker:

- ▶ Overview of Web Services Security in WebSphere Application Server 6.1 in the “Securing Web services applications using JAX-RPC at the message level” section (and its sub-sections) of the WebSphere Application Server 6.1 Information Center

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/twbs_securev6wss.html

- ▶ Reference information for configuring WS-Security in WebSphere Application Server 6.1 in the “Securing messages using JAX-RPC at the request and response generators” section (and its subsections) of the WebSphere Application Server 6.1 Information Center

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/twbs_secmesrrrg.html

- ▶ Overview of Web Services Security in WebSphere Message Broker in the “WS-Security” section (and its subsections) of the WebSphere Message Broker 6.1 Information Center

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp?topic=/com.ibm.etools.mft.doc/ac55630_.htm

- ▶ Reference information for configuring WS-Security in WebSphere Message Broker in the “Implementing WS-Security” section of the WebSphere Message Broker 6.1 Information Center

http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/topic/com.ibm.etools.mft.doc/ac60160_.htm

To test WS-Security configuration for a SOAP/HTTP Web service, use the following guidance:

- ▶ Use Rational Application Developer 7.0 or WebSphere Integration Developer 6.1 to generate Web service test client applications with the *Web service sample JSPs* interface from the target Web service.
- ▶ Use Rational Application Developer 7.0 or the WebSphere Integration Developer 6.1 *Secure Web Service Client* wizard to configure WS-Security for the generated test client “Based on a Secured Web Service.”
- ▶ Deploy the Web service and generated test client to the Rational Application Developer or WebSphere Integration Developer built-in runtime environment (WebSphere Application Server 6.1 or WebSphere Process Server 6.1).
- ▶ Use the TCP/IP monitor supplied with Rational Application Developer and WebSphere Integration Developer to monitor the HTTP port on which the Web service is listening.

- Use the test client to invoke a request, check the response, and view the TCP/IP monitor output to ensure that the Web service messages have been secured as expected.

Algorithms

Configuration of Web Services Security involves specifying algorithms for how the information is to be signed or encrypted. The Web service and its client must specify the same values for the algorithms, which is more complex in a scenario where a client and service are hosted by different products.

Consider the following examples:

- WebSphere Application Server, WebSphere Process Server, and WebSphere Enterprise Service Bus (WebSphere ESB) require a user to select individual algorithms for each appropriate part of the binding definitions.
- WebSphere Message Broker requires a user to select a single algorithm suite.

The algorithm suite *TripleDesRsa15* in WebSphere Message Broker matches the default values for algorithms that are set for WebSphere Application Server-based products. Table B-1 provides details.

Table B-1 Default WS-Security algorithms used in WebSphere Application Server bindings

| WebSphere Application Server Binding definition | Algorithm type | Algorithm value |
|---|-----------------------------------|---|
| Encryption Information | Data encryption algorithm | http://www.w3.org/2001/04/xmlenc#tripledes-cbc |
| | Key encryption algorithm | http://www.w3.org/2001/04/xmlenc#rsa-1_5 |
| Signing Information | Signature method algorithm | http://www.w3.org/2000/09/xmldsig#rsa-sha1 |
| | Canonicalization method algorithm | http://www.w3.org/2001/10/xml-exc-c14n# |
| Signing Information > Part Reference | Digest method algorithm | http://www.w3.org/2000/09/xmldsig#sha1 |
| Signing Information > Transform | Transform algorithm | http://www.w3.org/2001/10/xml-exc-c14n# |

For more details about algorithm suites, see “Web Services Security Policy Language (WS-SecurityPolicy),” July 2005 Version 1.1, on the Web at the following address:

<http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

LGI scenario implementation information

The following diagrams summarize the WS-Security configuration and application-level binding definitions that are created to sign and encrypt the Web service requests for the following reasons:

- ▶ Business process flow Web service import (Web service client to the MVR Check) using WebSphere Integration Developer
- ▶ MVR Check and Underwriter Web services using Rational Application Developer

Note: The definitions for response are the reverse of those shown for request.

Figure B-8 shows the application-level binding definitions that are created to sign a Web service request in WebSphere Application Server-based products.

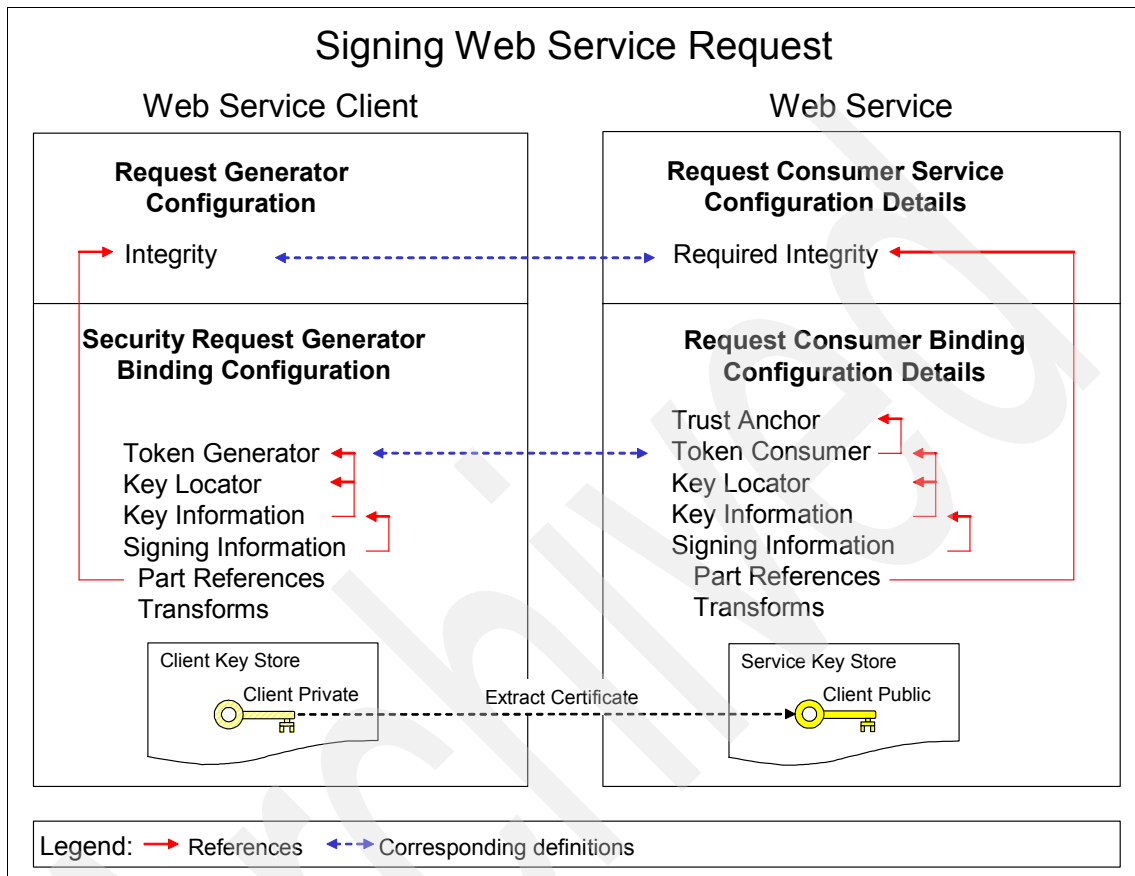


Figure B-8 Application-level binding definitions to sign a Web service request (WebSphere Application Server)

Figure B-9 shows the application-level binding definitions that are created to encrypt a Web service request in WebSphere Application Server-based products.

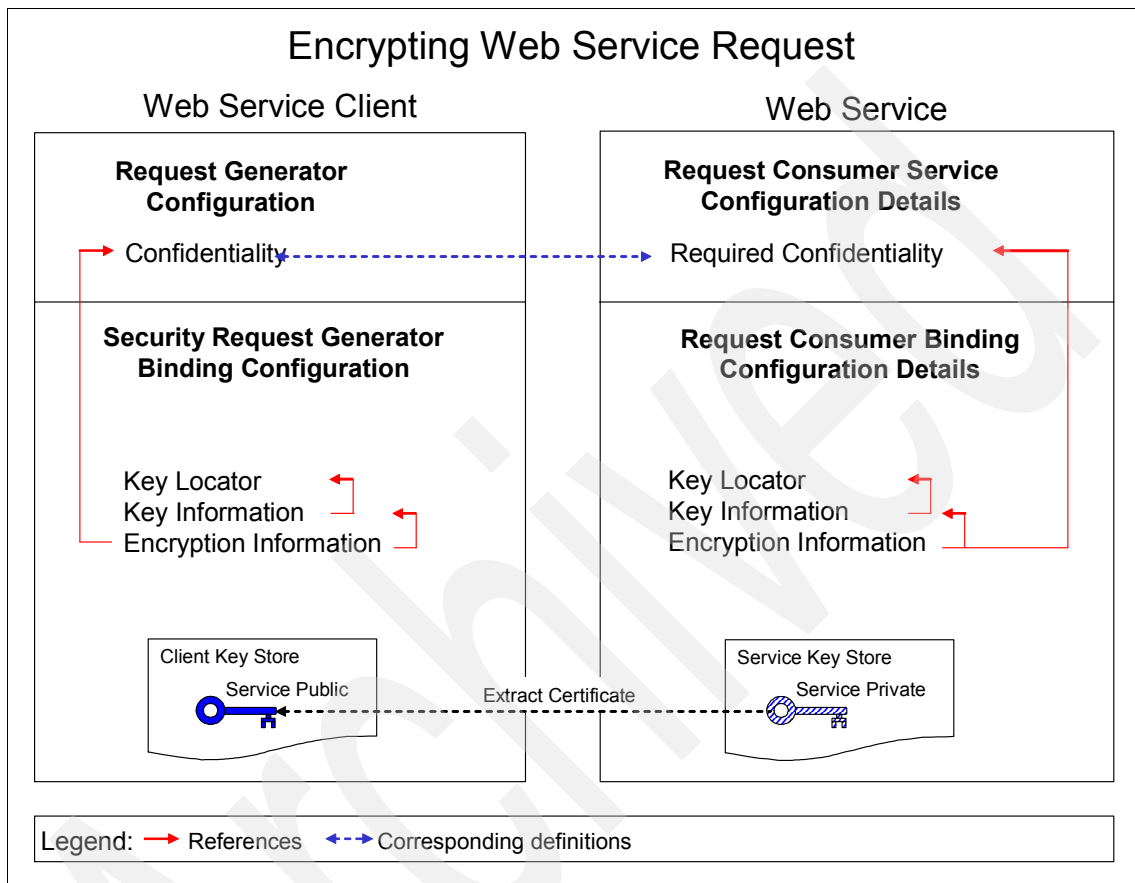


Figure B-9 Application level binding definitions to encrypt Web service request (WebSphere Application Server)

The following series of windows shows the definitions that are created for the WebSphere Message Broker Policy Set and Binding for the Web service message flow that acts as a consumer of the Underwriter Web service.

Figure B-10 shows how to enable message level protection.

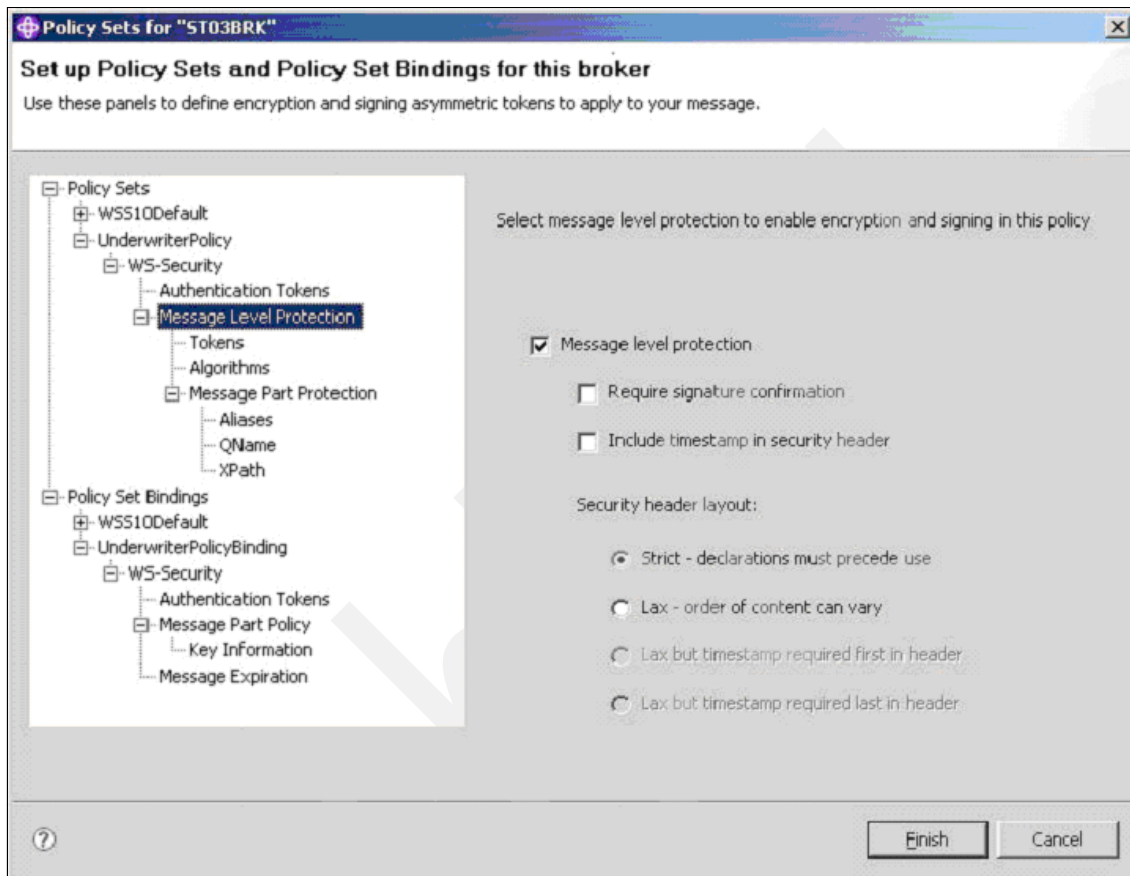


Figure B-10 WebSphere Message Broker Policy Set - Enabling Message Level Protection

Figure B-11 shows how to create tokens to sign and encrypt messages.

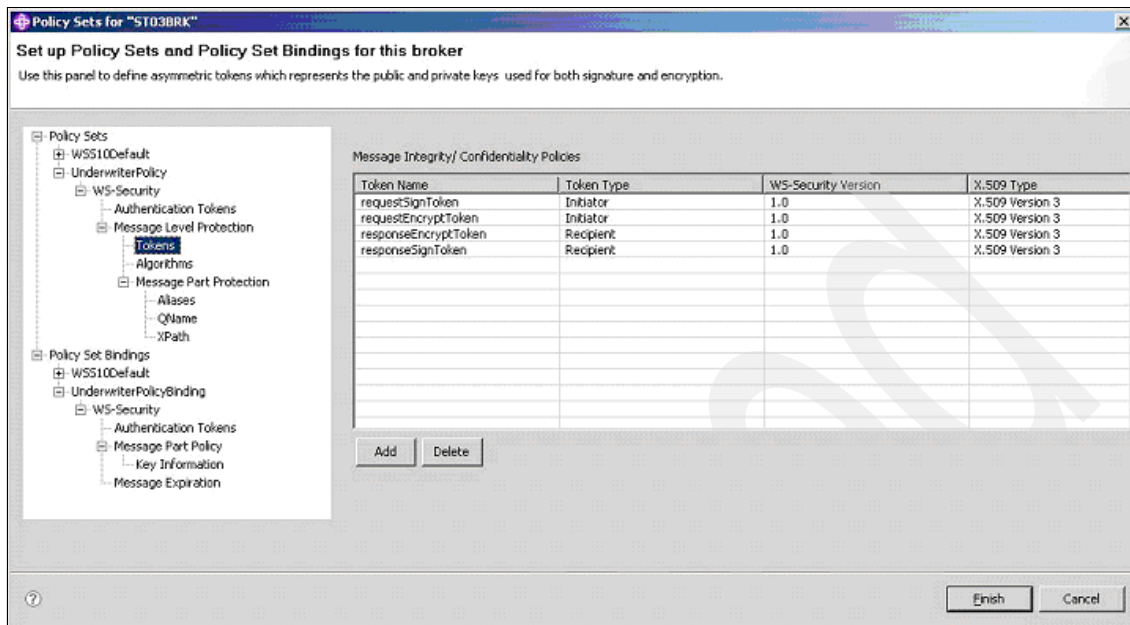


Figure B-11 WebSphere Message Broker Policy Set - Creating tokens for signing and encrypting messages

Figure B-12 shows how to select an algorithm suite for signing and encryption.

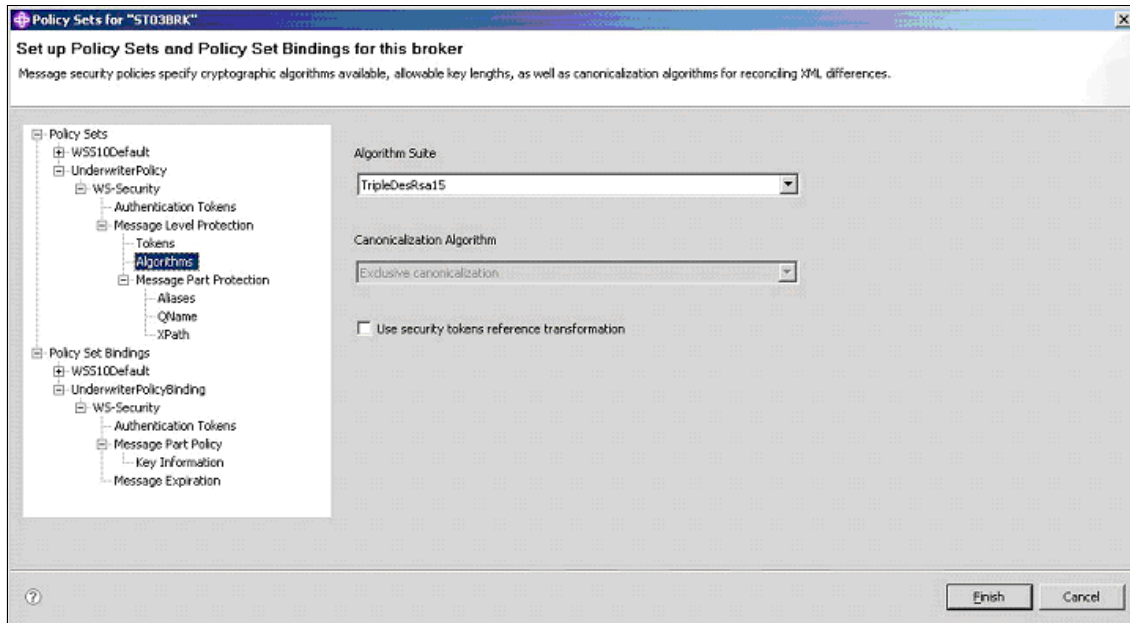


Figure B-12 WebSphere Message Broker Policy Set - Selecting an algorithm suite for signing and encryption

Policy Sets for "WS03BRK"

Set up Policy Sets and Policy Set Bindings for this broker

Use these panels to define the parts of your message to be encrypted and signed.

Policy Sets

- WSS10Default
 - UnderwriterPolicy
 - WS-Security
 - Authentication Tokens
 - Message Level Protection
 - Tokens
 - Algorithms
 - Message Part Protection**
 - Aliases
 - QName
 - XPath

- Policy Set Bindings
- WSS10Default
 - UnderwriterPolicyBinding
 - WS-Security
 - Authentication Tokens
 - Message Part Policy
 - Key Information
 - Message Expiration

Names with Security Type, SOAP Message and Message Body

| Name | Security Type | SOAP Message | Message Body |
|-----------------|---------------|--------------|--------------|
| requestSign | Signature | Request | Yes |
| responseSign | Signature | Response | Yes |
| requestEncrypt | Encryption | Request | Yes |
| responseEncrypt | Encryption | Response | Yes |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Add Delete

Finish Cancel

246 The Mixed Platform Stack Project

Figure B-14 shows how to create and associate the binding with the Policy Set.

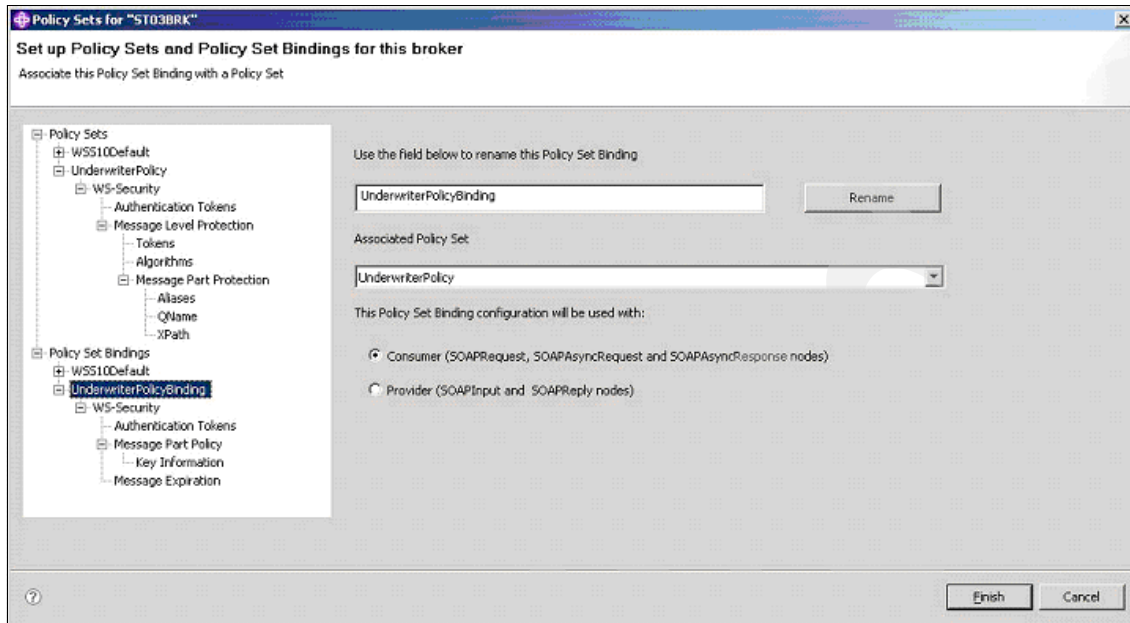


Figure B-14 WebSphere Message Broker Policy Set - Creating and associating the binding with the Policy Set

Figure B-15 shows how to associate message part tokens for the binding.

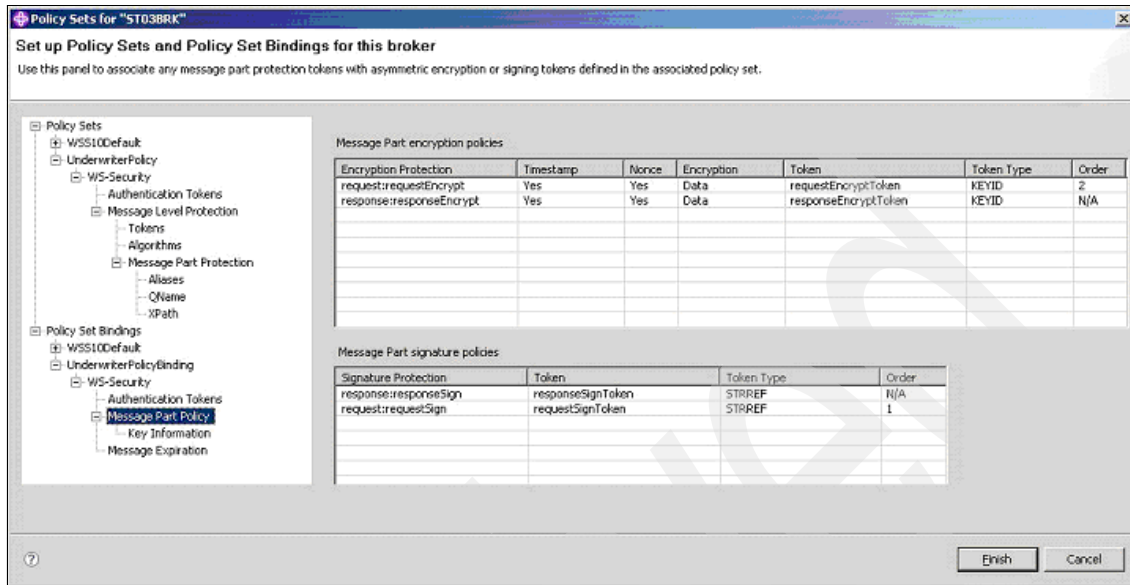


Figure B-15 WebSphere Message Broker Policy Set - Associating message parts with tokens for the binding

Figure B-16 shows how to associate tokens with certificates for the binding.

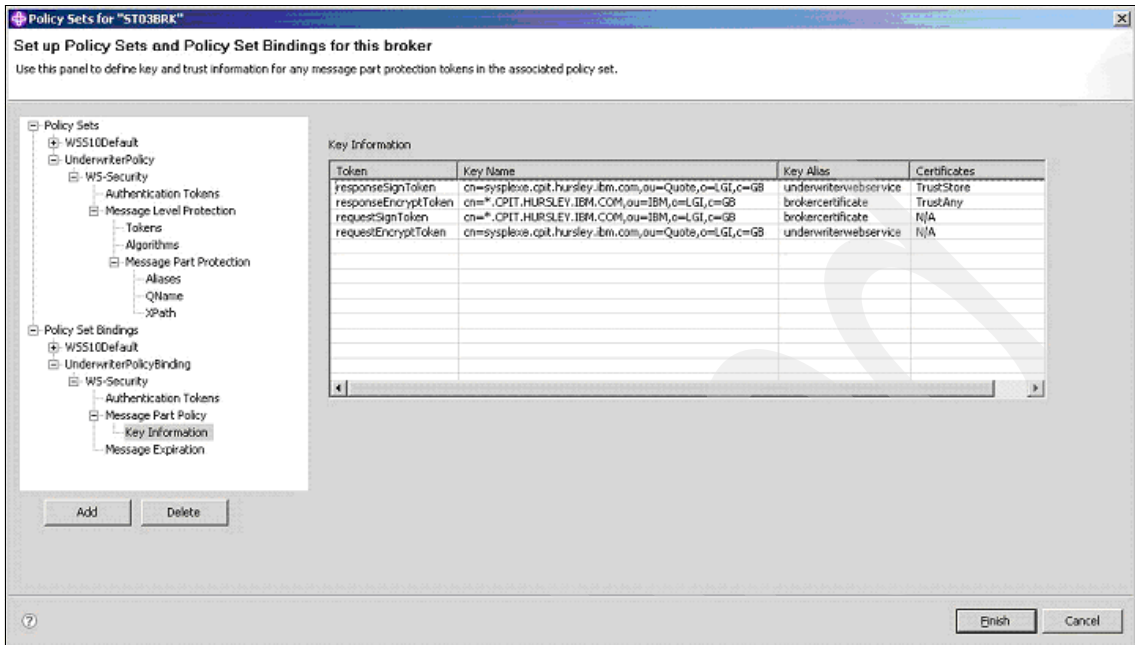


Figure B-16 WebSphere Message Broker Policy Set - Associating tokens with certificates for the binding

The requestSignToken and responseEncryptToken both reference the Web service client's personal certificate. The requestEncryptToken and responseSignToken both reference the Web service's public certificate.

Application deployment

This section provides an overview of the steps for deploying a Web service and the customization that is required for the target environment.

Deploying Web services and clients

Deploying Web services and clients with WS-Security configured to WebSphere Application Server or WebSphere Process Server involves the following steps:

1. Creating keystores and personal certificates in the target runtime environments for use by the Web service and client and exchanging the public certificates
2. Deploying the Web service and Web service client by using the administrative console

3. Creating or customizing WS-Security bindings for the target environment by using the administrative console with either of the following two options:
 - If using application-level bindings, the values coded within the application at configuration time might need to be customized through the “Web services: Client security bindings” and “Web services: Server security bindings” panels
 - If using server- or cell-level bindings, they must be defined.

For more information, consult the following resources:

- ▶ “Modify the application-level configurations in the administrative console” information in the “Securing Web services applications using JAX-RPC at the message level” topic of the WebSphere Application Server 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/twbs_securev6wss.html
- ▶ “Default Web services security configuration for details” topic in the WebSphere Application Server Toolkit Information Center
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.etools.webservice.security.doc/topics/cwbs_defaultconfigwsssecurity.html

Table B-2 summarizes the environment specific information that is typically in WS-Security bindings.

Table B-2 Environment specific information in WS-Security bindings

| Binding | Definition | Value |
|---|-----------------------------------|--|
| Web service (client) Request Generator | Token Generator & Call Handler | Web service client's keystore and certificate to be used for signing |
| | Key Locator & Key | Web service client's keystore and certificate to be used for signing |
| | Key Locator & Key | Web service client's keystore and Web service's public certificate to be used for encryption |
| Web service (client) Response Consumer | Trust Anchor | Web service client's keystore containing the trusted public certificates of the Web service |
| | Key Locator & Key | Web service client's keystore and certificate for encryption |

| Binding | Definition | Value |
|--------------------------------|--------------------------------|--|
| Web service Request Consumer | Trust Anchor | Web services' keystore containing the trusted public certificate of the Web service client |
| | Key Locator & Key | Web service's keystore and private certificate for encryption |
| Web service Response Generator | Token Generator & Call Handler | Web service's keystore and certificate for signing |
| | Key Locator & Key | Web service's keystore and certificate for signing |
| | Key Locator & Key | Web service keystore and Web service client's public certificate for encryption |

Troubleshooting

In the event of runtime problems after configuring WS-Security, the following techniques can assist in resolving the problem:

- ▶ Ensure that the WS-Security binding definitions in both the client and service specify the correct certificate and keystore values for the environment on which they are deployed.
- ▶ Ensure that, if an enterprise application contains more than one Web service or client call, the WS-Security configuration and any application-level bindings in the deployment descriptor extensions are associated with the appropriate service reference.
- ▶ Confirm that the deployed versions of the Web service and its client contain corresponding WS-Security configurations. Consider the following examples:
 - Attempting to use a client with no WS-Security configured to invoke a service that is configured to require integrity or confidentiality on the request will be rejected.
 - Attempting to use a client with a WS-Security binding specifying a token generator of a different type to the token consumer in the corresponding service binding will fail.
- ▶ Use error messages, trace entries, or TCP/IP sniffers to diagnose a problem or isolate it to request or response, client or service. Consider the example where in WebSphere Application Server-based environments, enabling the following trace provides more information about the Web service calls:

```
com.ibm.ws.webservices.trace.MessageTrace=all
```

Look for the following entries which display the content of the SOAP/HTTP request and response:

- More detailed WebSphere Application Server WS-Security tracing is available by enabling the following setting:
`Web Services Security=all`
- For WebSphere Message Broker Web service consumers and providers, check stdout and stderr logs. On z/OS, these are part of the execution group job log, while on distributed platforms, these are in `$MQSI_WORKPATH/components/brokername/executionGroupUUID`.
- More detailed information from WebSphere Message Broker can be obtained by enabling a Java Secure Socket Extension (JSSE) trace by adding the following statement to the broker's ENVFILE (z/OS) or mqsiprofile (distributed platforms) and restarting the broker:
`-Djavax.net.debug=true`
- ▶ Modify the WS-Security configurations to isolate the cause of problems. For example, configure only integrity or only confidentiality.

Known issues

IBM service is currently investigating the following issues that were discovered in this area:

- ▶ “ClassCastException customizing WS-Security bindings in WebSphere Process Server 6.1”
- ▶ “No predefined keystores customizing WS-Security bindings in WebSphere Application Server 6.1”

For details, see Appendix A, “Open issues” on page 149.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 256. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Designing for Solution-Based Security on z/OS*, SG24-7344
- ▶ *Exploiting IBM System z in a Service-Oriented Architecture*, SG24-7651
- ▶ *IBM WebSphere Application Server V6.1 Security Handbook*, SG24-6316
- ▶ *Security in WebSphere Application Server V6.1 and J2EE 1.4 on z/OS*, SG24-7384
- ▶ *The Role of IBM System z In the Design of a Service-Oriented Architecture*, REDP-4190
- ▶ *The Value of the IBM System z and z/OS in Service-Oriented Architecture*, REDP-4152

Other publications

The publication *z/OS V1R10.0 Comm Svr: IP Configuration Guide*, SC31-8775-13, is also relevant as a further information source.

Online resources

IBM provides a variety of information resources to complement the material contained in this document. Use the following information to learn more about the IBM products and solutions referenced by this document as well as related product offerings:

- ▶ IBM DB2 Version 9.1 for Linux, UNIX, and Windows Information Center
<http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp>

- ▶ IBM DB2 Version 9.1 for z/OS Information Center
<http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.db29.doc/db2prodhome.htm>
- ▶ IBM HTTP Server Version 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.ihs.doc/info/welcome_ihs.html
- ▶ Tivoli Directory Server 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm.IBMD5.doc_6.1/welcome.htm
- ▶ WebSphere Application Server 6.1 Information Center
<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp>
- ▶ WebSphere Message Broker 6.1 Information Center
<http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r1m0/index.jsp>
- ▶ WebSphere MQ 6.0 Information Center
<http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp>
- ▶ WebSphere MQ 7.0 Information Center
<http://publib.boulder.ibm.com/infocenter/wmqv7/v7r0/index.jsp>
- ▶ WebSphere Process Server 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/dmndhelp/v6r1mx/index.jsp?topic=/com.ibm.websphere.wps.610.doc/welcome_top_wps.htm
- ▶ WebSphere Service Registry and Repository 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/sr/v6r1/index.jsp?topic=/com.ibm.help_swg.ic.doc/wsrr_homepage.htm
- ▶ The following topics in the WebSphere MQ 6.0 Information Center:
 - “WebSphere MQ Security Introduction”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzas.doc/sy10220_.htm
 - “Authority to administer WebSphere MQ on UNIX and Windows systems,”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzas.doc/sy10750_.htm
 - “setmqaut (grant or revoke authority),” which shows the authorities that can be given to the different object types
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.amqzag.doc/fa15980_.htm

- “Stopping your queue manager putting messages to remote queues”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzah.doc/qc11430_.htm
- “Working with WebSphere MQ SSL support”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqzas.doc/sy11550_.htm
- “Authority to administer WebSphere MQ on z/OS”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqzas.doc/sy10770_.htm
- “Working with the Secure Sockets Layer (SSL) on z/OS”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzas.doc/sy12440_.htm
- “Setting up SSL communications”
http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/topic/com.ibm.mq.csqzas.doc/sy11560_.htm
- ▶ “WebSphere MQ queue connection factory settings,” in the WebSphere Application Server 6.1 Information Center
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.WebSphere.nd.multipatform.doc/info/ae/ae/umj_pqcfm.html
- ▶ WebSphere MQ SupportPac MO04, which is a useful tool for generating required commands and instructions for a variety of WebSphere MQ SSL configurations
<http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24010367>
- ▶ “Take_control_of_your_MQ_applications”
ftp://ftp.software.ibm.com/software/integration/wsrr/Take_control_of_your_MQ_applications.pdf
- ▶ WebSphere MQ Supportpac MA93
http://www-01.ibm.com/support/docview.wss?rs=171&uid=swg24017518&loc=en_US&cs=utf-8&lang=en
- ▶ “Accessing WebSphere Service Registry and Repository using WebSphere Message Broker V6.1 nodes”
http://www.ibm.com/developerworks/websphere/library/techarticles/0806_crocker/0806_crocker.html

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

The Mixed Platform Stack Project

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



The Mixed Platform Stack Project: Deploying a Secure SOA Solution into z/OS and Mixed z/OS and AIX Environments



Redbooks®

Deploy solutions to a z/OS platform

Secure the integration points

Test the deployment

The IBM System z platform is the strategic core of business world wide. By using a realistic customer scenario, two IBM teams set out to demonstrate how to deploy the IBM service-oriented architecture (SOA) portfolio on IBM z/OS and on z/OS in partnership with additional platforms such as AIX and Linux for System z. The teams created the experience that is documented in this IBM Redbooks publication to explain the work that is required to create, deploy, and test the SOA solution on both z/OS and z/OS with additional platforms. The teams also performed extensive testing to verify the correct behavior of the platforms, products, and applications involved.

This Redbooks publication covers the product configuration that is necessary to build the SOA solution described in the project scenario. This book provides useful hints and tips that were discovered during the course of testing to ensure successful solution deployment. It also provides an extensive set of references to other documents that proved useful for building the solution.

This book is designed for IT professionals who are interested in creating an SOA solution either entirely on z/OS or on z/OS in conjunction with other platforms. Prior to reading this book, you must have basic knowledge of SOA solutions, z/OS or other platforms, and the SOA products running on those platforms.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks