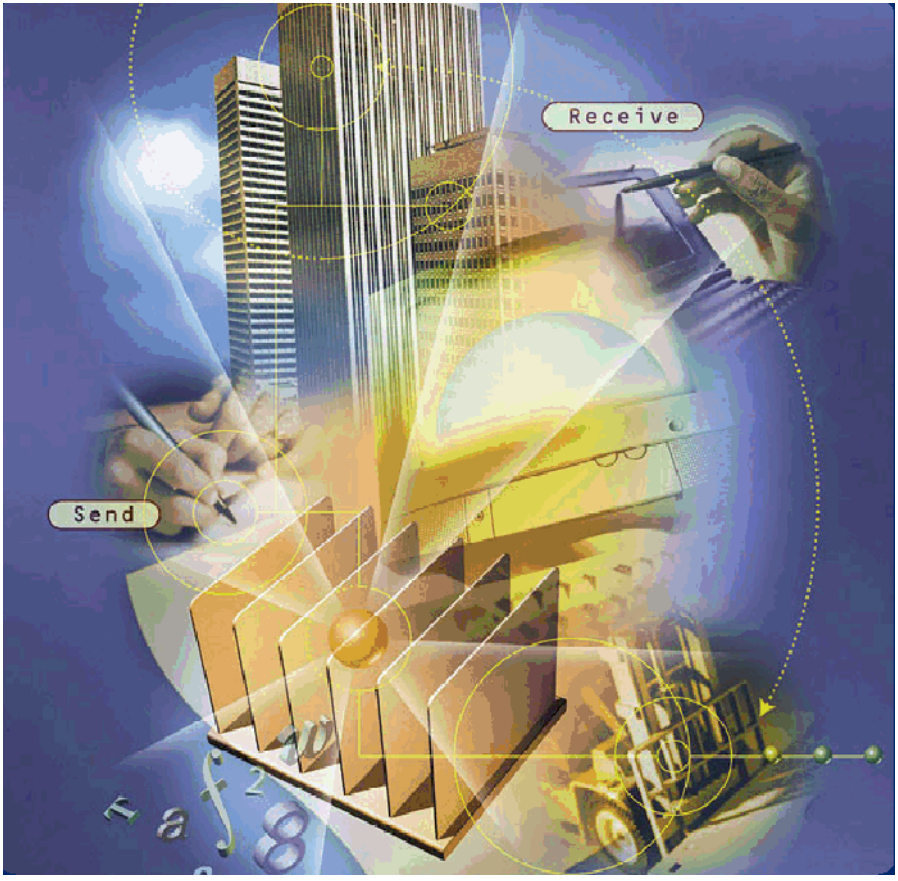
Progress®

SonicMQ™

SonicMQ Installation and Administration Guide



Printed in U.S.A.
November 2000

Contents

List of Figures

List of Tables

Preface

This Preface contains the following sections:

- "About This Manual" describes this manual and its intended audience.
- "Conventions in This Manual" describes the text formatting, syntax notation, and flags used in this manual.
- "Available Documentation" describes the printed and online documentation that accompanies SonicMQ.
- "Worldwide Technical Support" provides information on contacting technical support.

About This Manual

Progress SonicMQ is a fast, flexible, scalable E-Business Messaging Server designed to simplify the development and integration of today's highly distributed enterprise applications and Internet-based business solutions. SonicMQ is a complete implementation of the Java Message Service specification Version 1.0.2, an API for accessing enterprise messaging systems from Java programs.

This book is intended for software developers and system administrators. You can use this book for any of several purposes:

- Installing the SonicMQ product
- Configuring the SonicMQ product
- Configuring a third-party database for use with SonicMQ (guidelines)

- Administering the SonicMQ product

To make full use of this document, you should be familiar with:

- Installing applications on your supported platform
- Configuring Java applications
- Installing and configuring your third-party DBMS (if required)
- Basic concepts of messaging (if necessary, by first reading the *Getting Started with SonicMQ* guide)

This book consists of four chapters:

- Chapter 1, "Installation," tells you how to install and configure the SonicMQ messaging product to meet your needs.
- Chapter 2, "Administration Concepts," presents a conceptual overview of the tasks that are necessary for administering the SonicMQ messaging product.
- Chapter 3, "Graphical Administration Tool," describes Explorer, the graphical-user-interface administration tool that is part of the SonicMQ messaging product.
- Chapter 4, "Command-line Administration Tool," describes all the commands available through Admin Tool, the command-line administration tool that is part of the SonicMQ messaging product.

Note The Explorer described in Chapter 3 and Admin Tool described in Chapter 4 give you two ways to administrate SonicMQ. You can also administrate SonicMQ programmatically using the Management API, described in the *SonicMQ Programming Guide*.

Conventions in This Manual

In this section, you will find a description of the text-formatting conventions used in this manual and a description of notes, warnings, and important messages.

Typographical Conventions and Syntax Notation

This manual uses the following typographical conventions:

- **Bold typeface in this font** indicates keyboard key names (such as **Tab** or **Enter**) and the names of windows, menu commands, buttons, and other SonicMQ user interface elements. For example, "From the **File** menu, choose **Open**."

  Bold typeface is also used to highlight new terms when they are introduced in conceptual and overview sections.

- `Monospace typeface` is used to indicate text that might appear on a computer screen other than the names of SonicMQ user interface elements, including all of the following:

  – Code examples

  – Code that the user must enter

  – System output (such as responses, error messages, and so on)

  – Filenames and pathnames

  – Software component names, such as class and method names

  Essentially, `monospace typeface` indicates anything that the computer is "saying," or that must be entered into the computer in a language that the computer "understands."

  `Bold monospace typeface` is used to supply emphasis to text that would otherwise appear in `monospace typeface`.

  `Monospace typeface in italics` or ***`Bold monospace typeface in italics`*** (depending on context) indicates variables or placeholders for values you supply or that might vary from one case to another.

➤ **This symbol and font introduce a multi-step procedure:**

1. This is a first step.

   1.1 This is a step within a step.

2. This is a second step.

➤ **This symbol and font introduce a single-step procedure:**

□ This symbol starts a single-step procedure.

This manual uses the following syntax notation conventions:

- Where command-line examples are provided, a backslash character (\) indicates line continuation. It should not be entered on the actual command line.
- Brackets ([ ]) in syntax statements indicate parameters that are optional.
- Braces ({ }) indicate that one (and only one) of the enclosed items is required. A vertical bar (|) separates required items.
- Ellipses (. . .) indicate that you can choose one or more of the preceding items.

Note, Important, and Warning Flags

This manual highlights special kinds of information by using shading, placing horizontal rules above and below the text, and using a flag in the left margin to indicate the kind of information.

Note A **Note** flag indicates information that complements the main text flow. Such information is especially needed to understand the concept or procedure being discussed.

Important An **Important** flag indicates information that must be acted upon within the given context in order for the procedure or task (or other) to be successfully completed.

Warning A **Warning** flag indicates information that can cause loss of data or other damage if ignored.

Available Documentation

Table 1 lists the documentation supplied with SonicMQ. In addition to the documentation listed in this table, SonicMQ comes with sample files. All documentation is included with the SonicMQ media.

Table 1. The SonicMQ Documentation Set

Document	*Description*
SonicMQ Documentation Portal (`SonicMQ_Help.htm`)	Describes and links all SonicMQ online documentation components.
Getting Started with SonicMQ	Presents an introduction to the scope and concepts of the SonicMQ software and its packaging. Lists the features and benefits of SonicMQ in terms of its adherence to the Sun JMS specification and the extensions that make SonicMQ a richer, more useful messaging software.
SonicMQ Installation and Administration Guide	Describes configuration of various SonicMQ client types, clusters, and the message server and data stores. The administration chapters fully document server management using both the command-line interface and the graphical user interface administration tools. Covers security concepts and installation and administration of security features.
SonicMQ Programming Guide	Presents the SonicMQ sample applications and then shows how the programmer can enhance the samples, focusing on clients, connections, sessions, messages (including XML), transactions, and hierarchical topics.
SonicMQ Deployment Guide	The first part describes general deployment issues, including security. The second part concerns deployment issues for setting up dynamic routing for a B2B infrastructure.
SonicMQ API Reference	Contains information on the SonicMQ API that supplements the other manuals.
SonicMQ Product Update Bulletin	Describes enhancements to SonicMQ that are new with this release.
SonicMQ Release Notes	Provides late-breaking information and known issues.

Worldwide Technical Support

Progress Software's support staff maintains a wealth of information at http://www.sonicmq.com to assist you with resolving any technical problems that you encounter when installing or using SonicMQ Developer Edition.

From the SonicMQ home page, click on **Developer Exchange** to take advantage of resources for developers such as forums, downloads, tips, whitepapers, and code snippets.

For technical support for the SonicMQ Professional Developer Edition or the SonicMQ E-Business Edition, visit our TechSupport Direct Web page at http://techweb.progress.com. When contacting Technical Support, please provide the following information:

- The release version number and serial number of SonicMQ that you are using. This information is listed at the top of the Start Broker console window and might appear as follows:

  ```
  SonicMQ E-Business Edition [Serial Number 25677051]
  Release nnn Build Number nnn Protocol nnn
  ```

- Your first and last name.
- Your company name, if applicable.
- Phone and fax numbers for contacting you.
- Your e-mail address.
- The platform on which you are running SonicMQ, as well as any other environment information you think might be relevant.
- The Java Virtual Machine (JVM) you are using.

  To determine the JVM you are using, open a console window, go to the directory `SONICMQ_JRE` (default *`install-dir`*`\Java\bin`), and issue the command `.\jre -d`.

Table 2 provides information about Progress Software Corporation and its international offices.

Table 2. Progress Software International Offices

Locale, Office Name, and Address	*Contact Information*
North and Latin America: Progress Software Corporation 14 Oak Park Bedford, MA 01730 USA	**Pre-sales:** Telephone: 800 477 6473 ext. 4900 e-mail: `sonicmqpresales@progress.com` **Technical Support for Professional Developer Edition and E-Business Edition:** Telephone: 781 280 4999 Fax: 781 280 4543 e-mail: `support@progress.com`
Europe, the Middle East, Africa (EMEA): Progress Software Europe B.V. P.O. Box 8644 Schorpioenstraat 67 3067 GG Rotterdam THE NETHERLANDS	**Pre-sales:** e-mail: `sonicmqpresales-emea@progress.com` **Technical Support for Professional Developer Edition and E-Business Edition:** Telephone: 31 10 286 5222 Fax: 31 10 286 5225 e-mail: `emeasupport@progress.com`
Asia/Pacific: Progress Software Pty. Ltd. 1911 Malvern Road Malvern East, VIC Box 3145, AUSTRALIA	**Technical Support for Professional Developer Edition and E-Business Edition:** Telephone: 613 9885 0199 e-mail: `aussupport@melbourne.progress.com`

Chapter 1 Installation

In this chapter you will learn how to install SonicMQ and other software required to use the product effectively. A SonicMQ configuration consists of one or more **message servers** and one or more **clients**. Message servers are also called **servers** for short, or **brokers**.

Note Throughout this book, the terms broker and server are used interchangeably.

The chapter contains the following sections:

- "SonicMQ Editions" on page 20 tells where to find installation instructions for the different editions of SonicMQ.
- "Planning the Installation" on page 22 describes the software that is necessary or optional for use with SonicMQ Professional Developer Edition or E-Business Edition.
- "Installing the Server" on page 40 describes how to install a server for the SonicMQ Professional Developer Edition or E-Business Edition, including how to modify installation scripts and configuration files.
- "Starting and Stopping SonicMQ" on page 68 presents steps for starting a SonicMQ E-Business Edition server and application clients, including instructions on how to run SonicMQ as a Windows service.
- "Testing the Server" on page 74 describes how to test the server by running sample applications or by using Explorer.

- "Troubleshooting the Server Database" on page 74 tells what to do if your server cannot communicate with your database.
- "Installing Only the SonicMQ Client" on page 75 tells you how you can install a SonicMQ client without having to install the entire product.
- "Uninstalling SonicMQ" on page 77 tells you how to remove SonicMQ from your system.

If you are using a previous version of SonicMQ, see the *SonicMQ Product Update Bulletin* for information on upgrading your installation.

SonicMQ Editions

The editions of SonicMQ are:

- SonicMQ Developer Edition
- SonicMQ Professional Developer Edition
- SonicMQ E-Business Edition

All versions are contained on the installation CD. The version that you install depends on the license key you have. When you enter your license key during the installation process, you automatically obtain access to the proper version.

The E-Business and Professional Developer Edition offer the following features not found in the Developer Edition:

- Can be used with an unlimited number of clients
- Can support multi-server configurations for scalability
- Includes advanced security features

Full technical support is available for the Professional Developer Edition and E-Business Edition as described in "Worldwide Technical Support" on page 16.

The Developer Edition is restricted to a maximum of 100 concurrently connected clients connecting from at most one IP address other than the IP address of the server. There is no technical support for the Developer Edition, but registered users can find helpful information and technical advice at the **Developer's Exchange** at `http://www.sonicmq.com`.

The variations in the SonicMQ products are summarized in Table 3 and in the following two sections, which list the available security features.

Table 3. SonicMQ Editions

Edition	*Security Features*	*Maximum Number of Clients*	*Support and Maintenance Available*	*Deployable*
Developer Edition	Basic	100 (at one IP address)	No	No
Professional Developer Edition	Advanced	Unlimited	Yes	No
E-Business Edition	Advanced	Unlimited	Yes	Yes

Basic Security Features

All editions of SonicMQ offer a suite of basic security features:

- Challenge/response protocol for user authentication
- Message payload encryption (40-bit MD5/DES)
- Access Control Lists on topics and queues for users and groups
- Message digests to ensure integrity

Advanced Security Features

SonicMQ Professional Developer Edition and E-Business Edition offer a suite of advanced security features:

- All basic security features, except that the message payload encryption is increased to 56-bit MD5/DES
- Secure Socket Layer (SSL) protocol support—up to 128-bit encryption
- SSL supported for server-to-server communications, with certificate-based mutual authentication
- Certificate identity can be used as SonicMQ username
- SSL forward proxies supported
- Client-to-server SSL mutual authentication

Planning the Installation

Before installing SonicMQ, you must consider the following issues:

- Platform requirements
- Software compatibility
- Database options
- Multi-server installation
- Security
- ActiveX/COM clients

These issues are covered in the following sections.

Platform Requirements

Typical platform requirements are summarized in Table 4.

Table 4. Platform Requirements

Software Version	*OS*	*Hardware*		*Hard Disk Use*	
		CPU	*RAM*	*Minimum*	*Typical*
Developer Edition	Microsoft Windows	Pentium	64MB	100MB	200MB
	Sun Solaris	SPARC	64MB	100MB	200MB
E-Business Edition and Professional Developer Edition	Microsoft Windows	Pentium	256MB	300MB	1GB
	Sun Solaris	SPARC	256MB	300MB	1GB

The numbers given in Table 4 are guidelines only. The amount of RAM and hard disk space required depends on your disk cluster size, anticipated message traffic volume, the size of your messages, the number of other applications running, and the number of servers running.

See the Release Notes and `http://www.sonicmq.com` for a complete list of supported platforms.

Warning ***DO NOT*** use disks configured to use a write cache. This can lead to failure to recover messages in the event of a broker failure. Reliability cannot be guaranteed if you use disks configured to buffer writes.

Software Downloads

The Developer Edition is available as a free download from `http://www.sonicmq.com`. The size of the download is approximately 28MB for UNIX or Linux and 22MB for Windows. The ActiveX/COM client for SonicMQ is available as a separate download from the same location. The size of the ActiveX/COM client download is approximately 625KB. You can also download a C client product. The download is 682KB for Windows and 1MB for Solaris.

Ancillary Software

SonicMQ is designed to be used with certain third-party software. Some of this software is included with the product and some is not.

Software Included

By default, SonicMQ installs the IBM Runtime Environment for Windows®, Java™ Technology Edition, on Windows. For Solaris, it includes a JRE from JavaSoft which you must install manually.

Other JREs are included but you must install them manually.

SonicMQ ships with Cloudscape™, a 100% pure Java SQL database management system, embedded into the software.

The IBM XML Parser for Java Edition is included.

The SonicMQ Professional Developer and E-Business Editions ship with BSAFE-SSL-J from RSA Security.

Note See the *Release Notes* for the exact versions of included software.

Software Not Included

You can use an external database rather than the one that is embedded in SonicMQ as your persistent message store. Progress SonicMQ currently supports Oracle 8i and Microsoft SQL Server 7 databases. See "Database Options" on page 24 for details.

If you want to use JNDI with the JMS Administered Objects, you must obtain a JNDI service implementation from a third party. Get the JNDI service provider JAR files and include them in the CLASSPATH of the clients that use them (including the Explorer and Admin Tool administrative clients).

For the Professional Developer Edition and E-Business Edition you can use the IAIK SSL implementation, instead of using the BSAFE-SSL-J from RSA Security which is bundled with SonicMQ for SSL support. See "Using IAIK SSL" on page 30 for instructions on how to configure IAIK SSL.

Database Options

SonicMQ includes the Cloudscape database as an embedded data store. You can, however, use one of the two supported external JDBC-compliant databases:

- Oracle 8i
- Microsoft SQL Server Version 7

The database is used to hold security information, configuration information for multi-server installations, messages that are not immediately deliverable, and other information necessary for the reliable functioning of SonicMQ.

Unless you use the embedded database (configured during the installation process) you must configure your database to work with SonicMQ, and you must configure the SonicMQ server to find the database.

SonicMQ and Database Character Sets

SonicMQ stores message data in **Blob** fields. Other field types used by SonicMQ include **char**, **varchar2**, and **long**. SonicMQ is 100% Java, and Java completely supports Unicode, depending on the type of message that might get written to the database.

It is important to note that each database must be installed and set up to use the character set (code page) that is appropriate for the language used in the messaging application.

Depending on the database vendor, character sets can be chosen at install time and exist for every database created under that particular installation. For many databases, the character set chosen when the database is created cannot be changed without re-creating the database. Therefore the database character set chosen should always be a superset of the desired language or the equivalent of the messaging client's operating system's native character set.

Consult the database vendor's documentation for details on character sets and national language support.

Database Checklist

➤ **To deploy your external database for use with SonicMQ:**

1. If you are not using the default embedded database, you must set all properties in the `broker.ini` file that apply to the Database Information Function listed in Table 8 on page 50. These properties are:
   - `DB_CONNECT`
   - `DB_PASSWORD`
   - `DB_PROPERTIES`
   - `DB_USER`
   - `JDBC_DRIVER`
2. If you are using the Oracle 8i database, you must set the `SONICMQ_DBCLASS` environment variable in `startbr.bat` (on Windows) or `startbr.sh` (on UNIX or Linux) to point to the class library or zip file that contains the JDBC driver.
3. Make sure the `DB_PROPERTIES` property described on page 55 points to the `cfg` file that SonicMQ provides for your database (see "Database Configuration Files" on page 49).
4. Run `dbtool` as described in "Database Table Creation and Deletion Scripts" on page 47 to create the tables for your message server.

Planning for Security

SonicMQ security includes identification and authentication, access control, and encryption at the message level. These topics are covered in "Security" on page 86. In addition, the SonicMQ Professional Developer and E-Business Editions provide an SSL security package for encryption at the connection level, giving the highest degree of security. In this section, you will find instructions for customizing the SSL security features of SonicMQ.

Security is both an installation issue and a deployment issue. For information about deployment aspects of security, see the "Security" chapter in *SonicMQ Deployment Guide*. You will find information on topics that include:

- Firewall architecture
- HTTP Tunneling
- Forward and Reverse Proxies
- Use of Signed Applets
- Certificate-based Mutual Authentication
- Server initialization file encryption

You should only use the security appropriate to your application because encryption uses substantial resources. Message-level encryption degrades performance somewhat, and connection-level encryption (SSL) degrades performance substantially.

Enabling Security

To enable security you must set a configuration property and create the proper database tables. Then, after starting a server, you can use the supplied administrative tools to set up security options.

Warning Enabling security requires that you re-create the basic database tables, which means that you will lose any data stored in these tables. To avoid losing data, you should decide when first installing the server if you need to security-enable the server. If you do, you should enable security before using the server.

➤ **To enable security:**

1. Install the server, as described in "Basic Installation" on page 40 or "Installing Multiple Servers" on page 43.
2. Set `ENABLE_SECURITY=TRUE` in the `broker.ini` file. See "Server Initialization File" on page 49 for information on this file.
3. Re-create the basic database tables for the server as described in Table 7, "Use of dbtool" on page 48.
4. Create the security database tables, as described in "Database Table Creation and Deletion Scripts" on page 47.
5. After starting the servers, start SonicMQ Explorer (See Chapter 3, "Graphical Administration Tool") where you can create security-related items including:
   - Users (See "Users" on page 119.)
   - Groups (See "Groups" on page 125.)
   - Access Control Lists for Topics (See "Topics" on page 127) and Queues (See "Queues" on page 131.)
   - Quality of Protection for Topics (See "Topics" on page 127) and Queues (See "Queues" on page 131.)

   See "Security" on page 86 for a conceptual overview of security administration.

Note If you prefer a command-line interface to a GUI interface, you can use the Admin Tool (See Chapter 4, "Command-line Administration Tool") instead of Explorer. You can also use the Management API to administer the system programmatically.

Configuring the Default SSL

To use BSAFE-SSL-J from RSA Security, which is bundled with SonicMQ, follow the procedure below.

➤ To configure SSL:

1. Put the certificate chain and private key files (for example, PKCS12 or both PKCS7 and PKCS8) in *install-dir*/certs and trusted CA certificates (DER-encoded) in *install-dir*/certs/CA.
2. Edit the broker.ini file.

   2.1 Set:

   ```
   DEFAULT_SOCKET_TYPE=ssl
   ```

   2.2 Specify the server certificate chain and private key.

   To load the certificate chain and private key in PKCS12, set:

   ```
   SSL_CERTIFICATE_CHAIN=certs/pkcs12_file
   SSL_CERTIFICATE_CHAIN_FORM=PKCS12
   SSL_PRIVATE_KEY_PASSWORD=password
   ```

   To load the certificate chain in PKCS7 and the corresponding private key in PKCS8, set

   ```
   SSL_CERTIFICATE_CHAIN=certs/pkcs7_file
   SSL_CERTIFICATE_CHAIN_FORM=PKCS7
   SSL_PRIVATE_KEY=certs/pkcs8_file
   SSL_PRIVATE_KEY_PASSWORD=password
   ```

   2.3 Specify the cipher suite. You can use any of the cipher suites listed in "SSL_CIPHER_SUITES" on page 65. For example:

   To use strong encryption you might set:

   ```
   SSL_CIPHER_SUITES=SSL_RSA_WITH_3DES_EDE_CBC_SHA
   ```

   To use weaker encryption you might set:

   ```
   SSL_CIPHER_SUITES=SSL_RSA_EXPORT_WITH_RC4_40_MD5
   ```

To support several encryption methods, separate them with commas, for example:

```
SSL_CIPHER_SUITES=SSL_RSA_WITH_3DES_EDE_CBC_SHA,
SSL_RSA_EXPORT_WITH_RC4_40_MD5
```

2.4 Enable the server for SSL Mutual Authentication by setting:

```
SSL_CLIENT_AUTHENTICATION=TRUE
```

2.5 Specify the location of the CA certificates by setting:

```
SSL_CA_CERTIFICATES_DIR=certs/CA
```

2.6 If you are using server-to-server communication and this server is located behind a proxy, specify the proxy host and port:

```
TUNNELING_PROXY_HOST=proxy_hostname
```

```
TUNNELING_PROXY_PORT=proxy_port
```

3. Modify the startup scripts to enable a client to present a certificate for PKI Mutual Authentication.

If a client connects to a server that has the SSL peer authentication enabled, it must present its own certificate to connect to the server. For example, suppose you want to connect the administration clients Explorer and Admin Tool. On Windows, edit `explorer.bat`, `admin.bat`, and `SonicMQ.bat`. On UNIX or Linux, edit `setenv`. In either case, set system properties much as in the following examples where \ indicates line continuation:

```
set SONICMQ_SSL_CLIENT= \
-DSSL_CA_CERTIFICATES_DIR=certs/CA \
-DSSL_CERTIFICATE_CHAIN=certs/pkcs7_file \
-DSSL_CERTIFICATE_CHAIN_FORM=PKCS7 \
-DSSL_PRIVATE_KEY=certs/pkcs8_file \
-DSSL_PRIVATE_KEY_PASSWORD=password \
-DSSL_CIPHER_SUITES=SSL_RSA_WITH_3DES_EDE_CBC_SHA
```

To use PKCS12 instead of PKCS7 and PKCS8, replace:

```
-DSSL_CERTIFICATE_CHAIN=certs/pkcs7_file \
-DSSL_CERTIFICATE_CHAIN_FORM=PKCS7 \
-DSSL_PRIVATE_KEY=certs/pkcs8_file
```

with

```
-DSSL_CERTIFICATE_CHAIN=certs/pkcs12_file \
-DSSL_CERTIFICATE_CHAIN_FORM=PKCS12
```

Using IAIK SSL

To use the optional IAIK (Institute for Applied Information Processing and Communications) implementation of SSL, obtain the IAIK SSL libraries and copy them into your *install-dir*/lib directory. If you are running IAIK SSL with SonicMQ on Windows and using the IBM 1.1.8 JRE, you also need jdk11x_update.jar from IAIK.

Follow the procedure given in "Configuring the Default SSL" with the following changes:

- In the broker.ini file add the line:

  ```
  SSL_PROVIDER_CLASS=progress.message.net.ssl.iaik.iaikSSLImpl
  ```

- Modify the server startup script, startbr.bat (Windows) or setenv (UNIX or Linux):
  - Replace jsafe.jar, sslj.jar, and certj.jar with iaik_jce_full.jar, and iaik_ssl.jar in the classpath. For example, you might have:

    ```
    SONICMQ_SSL_LIB=install-dir\servname\lib\iaik_jce_full.jar;
    install-dir\servname\lib\iaik_ssl.jar
    ```

  - If you are running SonicMQ on Windows using the IBM 1.1.8 JRE, add jdk11x_update.jar to the classpath.

- Specify DSSL_PROVIDER_CLASS in client startup scripts. The setting of SONICMQ_SSL_CLIENT (given in Step 3 in "Configuring the Default SSL" on page 28) changes to the following:

  ```
  set SONICMQ_SSL_CLIENT= \
  -DSSL_PROVIDER_CLASS=progress.message.net.ssl.iaik.iaikSSLImpl \
  -DSSL_CA_CERTIFICATES_DIR=certs/CA \
  -DSSL_CERTIFICATE_CHAIN=certs/pkcs7_file \
  -DSSL_CERTIFICATE_CHAIN_FORM=PKCS7 \
  -DSSL_PRIVATE_KEY=certs/pkcs8_file \
  -DSSL_PRIVATE_KEY_PASSWORD=password \
  -DSSL_CIPHER_SUITES=SSL_RSA_WITH_3DES_EDE_CBC_SHA
  ```

Ciphersuites and Certificates

You can add settings for security to the configuration file broker.ini in order to use ciphersuites, certificates, keys, and passwords other than the provided defaults.

The properties that govern SSL security begin with `SSL_` and are described in "Server Initialization File" on page 49.

SonicMQ includes a GUI-based Certificate Management Tool. See "Certificate Management Tool" on page 177 for a description of this tool.

Connections and Acceptors

An **acceptor** is an object that listens for clients or for other servers.

This section contains technical information about the use of acceptors in initiating and receiving connections. It contains material about routing connections and routing acceptors that is best understood after reading the *SonicMQ Deployment Guide*.

Connections and Network Types

SonicMQ uses connections for several different purposes:

- A **configuration connection** is a connection between a message server and its configuration server. The configuration server centrally manages many functions on behalf of the servers that connect to it including security, cluster configuration, and routing information.
- An **interserver connection** is a connection between a message server and its peers in a SonicMQ cluster. Servers connected through an interserver connection form a SonicMQ cluster.
- A **routing connection** is a connection between servers or clusters in a routing network. Servers connected through routing connections form a routing network. Routing networks are part of the Dynamic Routing Architecture discussed in detail in the *SonicMQ Deployment Guide*.
- A **client connection** is a connection between a client and a message server or cluster.
- An **administrative client connection** is a special kind of client connection used for Explorer, Admin Tool, or clients that use the Management API.

Simultaneous Multi-protocol Support

You can configure a server with one or more acceptors. For each acceptor, you designate which type of connection it can receive.

SonicMQ supports four acceptor types:

- TCP-based
- SSL-based with mutual authentication
- SSL-based with server authentication
- HTTP-based

You might need to support multiple acceptors for several reasons:

- You might need to support different protocols for different connections.

  For example, some clients might connect with the message server using TCP, while others connect using HTTP.

  In another common case, you might have servers in a cluster connecting with each other using TCP for performance, while having clients connect with the cluster using SSL for security reasons.

- You might want some acceptors to include mutual authentication while others do not.
- You might want to use different sets of certificates on different acceptors.

You define the default acceptor for a server in the `broker.ini` file using the properties:

`DEFAULT_SOCKET_TYPE=`*`socket_type`*

`PORT=`*`port_num`*

You can define additional acceptors in the `broker.ini` file by setting `NUM_ACCEPTORS=`*`num`*, where *num* > 1, and for each nondefault acceptor (*n* > 1) using the properties:

`SOCKET_TYPE_`*`n`*`=`*`socket_type`*

`PORT_NUMBER_`*`n`*`=`*`port_num`*

`IP_OR_HOST_n={`*`IP_address`*`|`*`host_name`*`}` (optional)

You must define additional acceptors prior to starting the message server. You cannot add them dynamically (while the message server is running).

See "Server Initialization File" on page 49 for details about the `broker.ini` properties mentioned above.

Configuring Acceptors on a Receiving Server

The same acceptor can be designated for several connections, possibly used for different purposes. By default, each server has at least one acceptor, the default acceptor. The following configuration properties designate certain acceptors for receiving specific kinds of connections:

`INTERBROKER_ACCEPTOR=`*`n`*

`DEFAULT_ROUTING_ACCEPTOR=`*`URL`*

See "Server Initialization File" on page 49 for details about these `broker.ini` properties.

Choosing Acceptors on a Connecting Client or Server

A client or connecting server chooses which acceptor to connect to on the receiving broker as follows:

- A client chooses the acceptor of the server it connects to programmatically. To establish a connection using a nondefault acceptor, you supply the server name, port, and protocol used by the acceptor when creating the connection.
- To connect to a configuration server, an initiating server determines the acceptor of the configuration server using the `IB_CONFIG_SERVER` property of the initiating server.
- For an interserver connection, the connecting server automatically chooses the acceptor specified by the `INTERBROKER_ACCEPTOR` property of the receiving server.
- For a routing server you define a routing connection to another routing server or cluster through an administrative tool such as Explorer or Admin Tool. The routing connection information specifies which acceptor to connect to on the receiving server when initiating a new routing connection.

If a routing connection that has been established to the current server by another server times out or fails, the current server will not use the routing connection information. Instead, it will try to reconnect to the other server to resolve indoubt messages using the acceptor specified by the other server in its DEFAULT_ROUTING_ACCEPTOR property.

Load Balanced Connections Within a Cluster

Servers within a SonicMQ cluster can load balance an incoming connection from a client or a connecting routing server by redirecting the connect request to another server within the same cluster. Load balancing can only redirect connections to another server's default acceptor. Thus, clients or connecting servers requesting a load balanced connection to a cluster might have their connection redirected to a default acceptor of one of the cluster servers.

Important Each acceptor **must** have a unique port number and each must be a port not currently in use by any other applications.

Installing the ActiveX/COM Client

The SonicMQ ActiveX/COM client is only supported on Windows. It uses the JavaSoft JavaBeans ActiveX/COM Bridge, which requires that you have a specific JRE. Before installing the ActiveX/COM client, you must have the following installed:

- SonicMQ (See "Installing the Server" on page 40 for instructions.)
- Sun Java Runtime Environment (JRE), which is included with SonicMQ but must be installed separately. For the appropriate version for your platform, see the *SonicMQ Release Notes* or **http://www.sonicmq.com**.

The installation process checks for this required software and warns you and exits if it discovers that the required software is missing.

The ActiveX/COM client is available on the installation CD, or from **http://www.sonicmq.com** as a separate download.

➤ **To install the ActiveX/COM client:**

1. If you do not have an installation CD, download the SonicMQ ActiveX/COM client from `http://www.sonicmq.com`. If you have an installation CD, navigate to the `activex` directory.
2. Run `setup.exe` and follow the directions given by InstallShield.

➤ **To uninstall the ActiveX/COM client:**

1. Double-click on **Start > Settings > Control Panel > Add Remove Programs**.
2. Select **Progress SonicMQ 3.0 ActiveX** in the **Install/Uninstall** list.
3. Click **Add/Remove Programs**.
4. Confirm that you want to remove the program.

Deployment Scenarios

This section presents some of the ways that you can deploy SonicMQ:

- **Single-system one server deployment** — One server with clients on the same machine
- **Development deployment** — One server machine and multiple client machines
- **Multi-server single-machine deployment** — Multiple servers running as a cluster on the same machine
- **Basic production deployment** — Multiple servers running as a cluster on multiple machines

For deployment scenarios that involve routing networks, see the *SonicMQ Deployment Guide*.

Single-system One Server Deployment

To get up and running quickly with SonicMQ you can install a server and one or more clients on the same machine. This configuration is typically used in small-scale test scenarios. You can create this deployment with any edition of SonicMQ, although you are limited as to the number of clients you can support if you use the Developer Edition. You can use an unlimited number of clients with the Professional Developer or E-Business Editions. A schematic of the installation is shown in Figure 1.

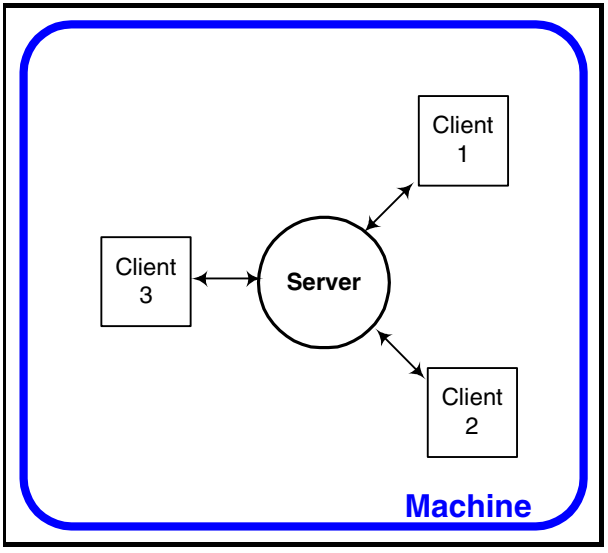


Figure 1. Sample Single-system One Server Deployment

Installing SonicMQ for a single-system one server deployment, see "Basic Installation" on page 40. Starting SonicMQ in a single-system one server deployment is covered in the first procedure under "Starting the SonicMQ Server" on page 68.

Development Deployment

The development deployment, shown in Figure 2, is very similar to the single-system one server deployment described in the preceding section, with the exception that the clients are running on different machines. You can create this deployment with any edition of SonicMQ, although you are limited as to the number of clients you can support if you use the Developer Edition. You can use an unlimited number of clients with the Professional Developer Edition or E-Business Edition.

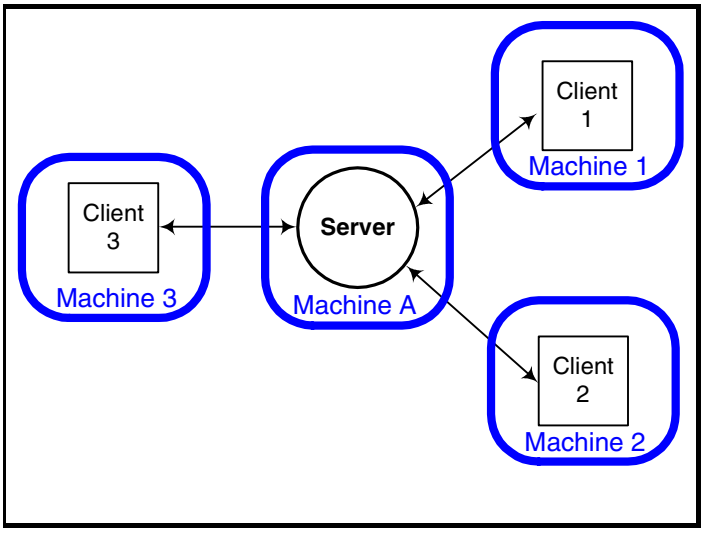


Figure 2. Sample Development Deployment

Installation of SonicMQ for a development deployment is covered in "Basic Installation" on page 40. Starting SonicMQ in a development deployment is covered in the first procedure under "Starting the SonicMQ Server" on page 68.

Multi-server Single-machine Deployment

SonicMQ support multi-server environments, using server clustering. In a production environment you would typically run each server on a separate machine. However, you might want to run all servers on one machine for testing purposes. (If you have the Developer Edition your entire configuration including servers and clients is limited to at most two separate machines, so if you want to test a cluster of more than two servers, you must place multiple servers on one machine.) This is called a **multi-server single-machine deployment**. A schematic of the installation is shown in Figure 3.

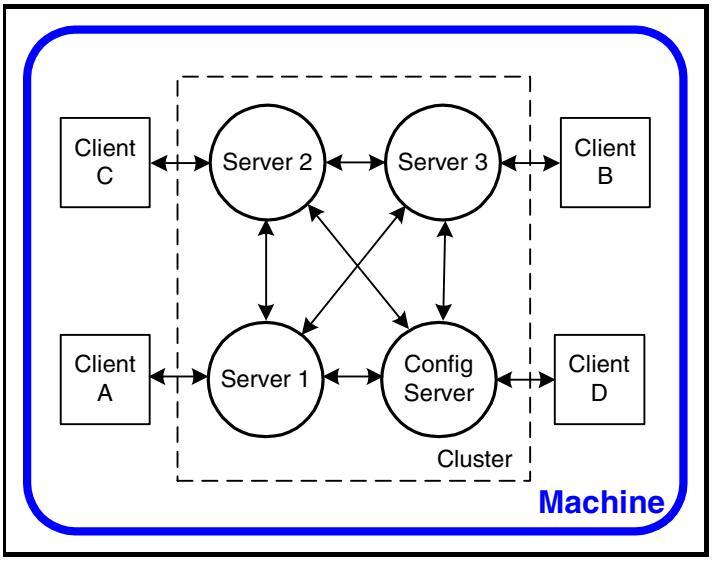


Figure 3. Sample Multi-server Single-machine Deployment

This configuration is only possible with the SonicMQ Professional Developer or E-Business Editions.

Installing SonicMQ in this configuration is covered in "Installing Multiple Servers" on page 43. Starting SonicMQ in a multi-server single-machine deployment is covered in the second procedure under "Starting the SonicMQ Server" on page 68. For instructions on configuring a cluster see "Clusters" on page 114 or "Server and Cluster Administration Commands" on page 198. Also see "Adding Servers to Security-enabled Clusters" on page 81.

Basic Production Deployment

For a production environment with large volumes of traffic, you should choose multiple servers, each running on its own machine. The production deployment shown in Figure 4 is similar to the multi-server single-machine deployment described in the preceding section, except that the servers are running on different machines. In this example, the configuration server is part of the cluster.

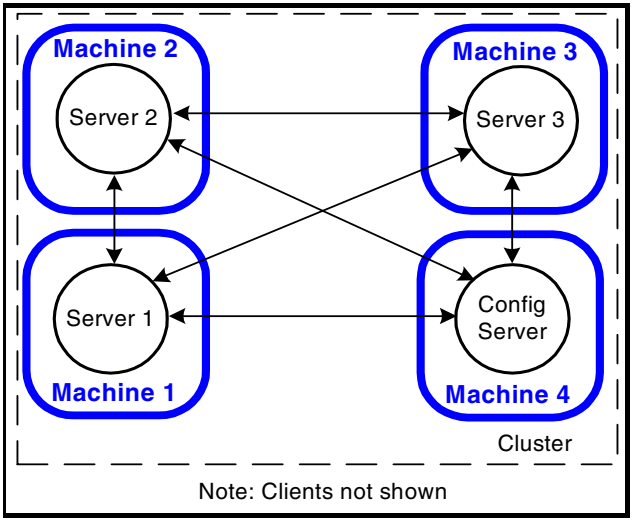


Figure 4. Sample Basic Production Deployment

This configuration is possible with the SonicMQ Professional Developer Edition or E-Business Edition. However, it cannot be used with the SonicMQ Professional Developer Edition in a production environment. You must give each server a separate name.

To start servers in a production environment, you will probably want to have the server start automatically with system startup. On UNIX or Linux, do this by placing `startbr.sh` in your startup script. On Windows, install SonicMQ as a Windows service, as explained in "Setting Up SonicMQ as a Windows Service" on page 70.

Installing the Server

The server is the heart of SonicMQ. Before clients can communicate with one another, at least one server must be installed, configured, and started. The following sections show you how to install and configure servers.

Basic Installation

➤ To install Progress SonicMQ on Windows:

1. Run setup from either a CD or a downloaded file:
   - From a CD, run:

     `D:\setup`

     where `D` is the drive letter for your CD-ROM drive.
   - From a download, run `setup.bat` from the directory in which it exists.
2. Follow the instructions in the installation wizard. If you want to use a nondefault JVM, you can specify the path to that JVM during the installation process.

See the *Getting Started with SonicMQ* guide for an overview of the product and instructions on running the samples.

➤ To install Progress SonicMQ on UNIX or Linux:

Note These instructions are Solaris-specific. Some steps might vary slightly for other versions of UNIX or for Linux. See your operating system manual for installing software.

1. Insert the CD-ROM into your machine's CD-ROM drive.
2. Change the directory to your cdrom directory. If you do not have a cdrom directory, or autodaemon (volume management daemon) is not running on your machine, mount the CD using the command:

   `mount -F hsfs -o ro,nrr -r device-name mount-point`
3. The InstallShield utility requires a JVM to run. Make sure you have a JVM installed and in your PATH.

4. Run the install on the console. If you are installing on a remote machine, set DISPLAY:

   ```
   DISPLAY=machine:0.0;export DISPLAY
   ```

5. Run the InstallShield startup script: `setup.sh`.
6. Follow the instructions in the installation wizard.
   - 6.1 When asked to choose a port, you should choose a number greater than 1024, as ports 1–1024 are reserved for the operating system.
   - 6.2 When asked to specify the path to the JVM you wish SonicMQ to use you may specify a JVM that is different from the one you are using to run InstallShield.
7. See the *Getting Started with SonicMQ* guide for an overview of the product and instructions on running the samples.

Unattended Installation

An **unattended installation**, also called batch or silent installation, presents no GUI and can be run unattended, so you can use it for remote installations or in a script so that many installations can be performed without user intervention. You configure the installation prior to running it by editing a setup file.

Unattended Installation Display Requirement

Even though an unattended installation does not present a GUI, your installation machine must still have display capabilities. This means that on a UNIX or Linux machine, the `DISPLAY` environment variable must be set.

Configuring the Setup File

The unattended installation requires a properly configured `setup.ini` file. The installation looks for `setup.ini` in the target installation directory specified from the command line at run time. If the installation fails to find `setup.ini` in this directory, it will look for it in the directory containing the `setup.class` file.

At deployment, `setup.ini` is located in the same directory as `setup.class`. If this directory is not on a CD and is writable, you can edit `setup.ini` in place. However, if `setup.class` is on a nonwritable medium and is run from this

location, you must copy `setup.ini` to the SonicMQ installation directory. Alternatively, you can copy the entire installation directory structure to a writable directory and install it from there.

Table 5 describes the properties you can set in `setup.ini`.

Table 5. Unattended Installation Setup Properties

Property	*Required*	*Default*
CONTROL_NUMBER	Yes	None
JVM_PATH	No	For Windows, the bundled JVM For UNIX or Linux, the JVM used to run `setup.class`
BROKER_NAME	No	SonicMQ
BROKER_PORT	No	2506
INSTALL_LOG_FILE_DIRECTORY	No	INSTALL_DIR

Unattended Installation Log File

Output from the unattended installation is sent to a log file, `silent.log`. By default, `silent.log` is in the target installation directory, `INSTALL_DIR`. You can change this in `setup.ini` by setting `INSTALL_LOG_FILE_DIRECTORY` to `CURRENT_DIR` or any other writable directory path.

Any error in installation produces a line beginning "`ERROR:`" in the log file.

If one or more errors are encountered, the last line of the log file will read:
`ERROR: Installation encountered errors.`

If no errors are encountered, the last line of the log file will read:
`SUCCESS: Installation complete.`

Running an Unattended Installation

➤ **To run an unattended installation on Windows:**

☐ Enter the command:

`setup [`*`path-to-java-exe`*`] -silent `*`install-dest`*

➤ **To run an unattended installation on UNIX or Linux:**

- ☐ Enter the command:

```
setup.sh -silent install-dest
```

In both commands, *path-to-java-exe* specifies the JVM from which to run the installation. This path must point to a directory containing `java.exe` or `jre.exe`. In both commands, *install-dest* is the directory where SonicMQ will be installed.

Installing Multiple Servers

With SonicMQ Professional Developer and E-Business Editions, you can install multiple servers and configure them to work together in clusters. In a production deployment, you would usually run the multiple servers on separate machines. However, for testing purposes or with multiple-CPU machines, you might want to run multiple servers on one machine.

One server serves as a configuration server, which maintains information about the cluster. This configuration server can itself be a member of a cluster, but it does not have to be.

Installing and Running Multiple Servers on Multiple Machines

To install multiple servers on multiple machines you must install the software on each machine. In performing the installations, you must give each server a different name.

1. Perform a normal installation of SonicMQ Professional Developer Edition or E-Business Edition on each machine.

Important

During each installation process, you must change the default server name so that it is different from those in the previous installations. For each installation, go to the installation directory and edit the `broker.ini` file:

- For each server that is to be in a cluster, you must have `ENABLE_INTERBROKER=TRUE`. If you want to enable security for the cluster, you must set `ENABLE_SECURITY=TRUE` and `BROKER_PASSWORD=`*`password`* for every server in the cluster. See "Configuring Servers for a Security-enabled Cluster" on page 89 for more information.
- For each server to be in a cluster, except the one you want to designate as the configuration server, you must have `IB_CONFIG_SERVER={tcp://|http://|ssl://}`*`hostname`*`:`*`port`*, where *`hostname`* is the machine hosting the configuration server and *`port`* is the port number reserved for the configuration server.

2. If you are enabling security, you must re-create the basic database tables for the configuration server. Go to the installation directory of the configuration server and:

   2.1 Run `dbtool /d basic` (Windows) or `dbtool -d basic` (UNIX and Linux) to delete the old tables.

   2.2 Run `dbtool /cs basic` (Windows) or `dbtool -cs basic` (UNIX and Linux) to create new tables (with sample queues)

   See Table 7, "Use of dbtool" on page 48 for more details and options.

3. Go to the installation directory of the configuration server and:

   3.1 If you are enabling security, initialize the security database tables:

   ➔ On Windows, run `dbtool /c security`

   ➔ On UNIX, run `dbtool -c security`

   3.2 Initialize the server cluster database tables:

   ➔ On Windows, run `dbtool /c interbroker`

   ➔ On UNIX, run `dbtool -c interbroker`

4. Start each server.
5. From any of the machines on which you have installed SonicMQ, start Explorer and:
   - 5.1 Connect to the configuration server as described in "Connecting to a Server" on page 110.
   - 5.2 Create one or more clusters as described in "Clusters" on page 114.

Installing and Running Multiple Servers on One Machine

Important The option of installing multiple servers on one machine lets you create test multi-server configurations. Installing multiple servers on one machine increases the complexity of your installation and yields no increase in performance unless the machine has multiple CPUs, so it is not recommended for production deployments with single-CPU machines.

The procedure for installing multiple servers on one machine is similar to the procedure for installing multiple servers on multiple machines. This section highlights the differences.

The easiest way to install multiple servers on one machine is to install multiple copies of the software. In performing the installation, you have to change the following values so that they are unique for each installation:

- Installation directory
- Server name
- Port number (for UNIX and Linux, this number must be greater than 1024)

In addition, the `LOG_PATH` value for each server must be different. If you use the default log path, this condition is satisfied automatically.

When you set `IB_CONFIG_SERVER={tcp://|http://|ssl://}`*`hostname`*`:`*`port`* in the `broker.ini` files for the cluster members, *`hostname`* can be `localhost`.

In a Windows environment, the **Start Broker** tool available through the Windows **Start** menu applies only to the server that was installed last. Every other server must be started by going to its installation directory and running `startbr.bat`.

Similarly, the **Uninstall** tool available through the Windows **Start** menu applies only to the server that was installed last. Every other server must be uninstalled by going to its installation directory and running `uninstall.bat`.

Scripts and Configuration Files

The behavior of SonicMQ is controlled by several files:

- Scripts (`bat` files for Windows or `sh` files for UNIX or Linux)
- Database configuration files (`cfg` files)
- Initialization files (`ini` files)

The installer creates backup copies of scripts and initialization files that it changes. On UNIX or Linux platforms, the installer creates backup copies of `broker.ini`, `startbr.sh`, and `setenv`. On Windows platforms, the installer creates backup copies of `broker.ini` and `startbr.bat`.

Available Scripts

SonicMQ uses a number of the scripts (`bat` files for Windows or `sh` files for UNIX or Linux) that are located in the `bin` subdirectory of the SonicMQ installation directory. Table 6 lists the scripts with a short description of each.

Table 6. SonicMQ Scripts

Windows Script	*UNIX Script*	*Description*
`admin.bat`	`admin.sh`	Starts the Admin Tool. (See Chapter 4, "Command-line Administration Tool.")
`explorer.bat`	`explorer.sh`	Starts Explorer. (See Chapter 3, "Graphical Administration Tool.")
`help.bat`	`help.sh`	Provides access to the Documentation portal.
`dbtool.bat`	`dbtool.sh`	Creates or deletes the database tables for the server. (See "Database Table Creation and Deletion Scripts" on page 47.)
`startbr.bat`	`startbr.sh`	Starts the server. (See "Starting and Stopping SonicMQ" on page 68.)
`uninstall.bat`	NONE	Runs the uninstall utility.

Database Table Creation and Deletion Scripts

SonicMQ ships with database table creation/deletion scripts: `dbtool.bat` for Windows or `dbtool.sh` for UNIX or Linux. You can use `dbtool` for any supported database.

Note The `createdb.bat` script in *`install-dir`*`\bin` is used solely to create the Cloudscape database at install time. Do not use it after the installation is complete. Instead, use dbtool for deleting, creating, and removing tables.

The syntax for the `dbtool` command is:
`dbtool` *`action tables`*

where *`action`* is one of:

- `/c` (Windows) or `-c` (UNIX and Linux) to create
- `/cs` (Windows) or `-cs` (UNIX and Linux) to create with sample queues
- `/d` (Windows) or `-d` (UNIX and Linux) to delete
- `/r` (Windows) or `-r` (UNIX and Linux) to re-create, that is to delete and then create

and *`tables`* is `all` or one or more of:

- `basic`
- `interbroker`
- `security`

These scripts set the path to the JVM and the CLASSPATH and then run a Java program that performs the initialization.

Typical ways of using `dbtool` are presented in Table 7.

Table 7. Use of dbtool

If You Want to...	*Windows Commands (for UNIX or Linux, replace / with -)*	*Comments*
Install one server using the embedded database, without security	No command is required.	Basic tables are automatically created in the default database when you install SonicMQ.
Create basic tables for a nondefault database	`dbtool /c[s] basic`	Use `cs`, not `c`, if you want to install the sample queues (recommended).

Create tables for a security-enabled server, using the default database (sample queues created)	`dbtool /d basic`
	`dbtool /cs basic`
	`dbtool /c security`
		Set `ENABLE_SECURITY=TRUE` in the `broker.ini` file before issuing the second command.

Create tables for a security-enabled server, using the default database (sample queues not created)	`dbtool /r basic` `dbtool /c security`	Set `ENABLE_SECURITY=TRUE` in the `broker.ini` file before issuing the first command.

Create tables for a security-enabled configuration server, using the default database (sample queues created)	`dbtool /d basic`
	`dbtool /cs basic`
	`dbtool /c security`
	`dbtool /c interbroker`
		Modify `broker.ini` files as described in Step 1 on page 43 for the configuration server and other servers in the cluster before issuing the second command.
		Do not create security or server cluster tables for the other servers in the cluster.

Warning When using the delete or re-create options, be careful that you do not inadvertently delete existing server tables.

Note Before running `dbtool`, change the directory to the `bin` subdirectory of the server installation directory.

The `cs` option can only be used with the basic server tables. If you re-create basic server tables using `dbtool /r basic` (Windows) or `dbtool -r basic` (UNIX and Linux), the new basic server tables will not contain the sample queues, even if the original ones did. (The four sample queues are required if you want to run the queue samples provided with SonicMQ.)

Database Configuration Files

SonicMQ supplies a database configuration file for each supported database type. These files have the `.cfg` suffix. You must have a copy of the database configuration file for your database in each of your server installation directories.

Explorer Initialization File

The SonicMQ Explorer administration tool uses the `explorer.ini` file to hold initialization information for the tool.

Server Initialization File

Each server you run must have a server initialization file, initially named `broker.ini`, that resides in the server installation directory. You can rename this file or encrypt it by using the PBETool, which is described in the *SonicMQ Deployment Guide*. A `broker.ini` file is created at installation. You might need to tailor it for your environment.

The `broker.ini` file sets a number of environment variables by means of lines of the form *`property=value`*.

The most commonly used properties are listed in Table 8, where they are grouped by function and followed by a pointer to a short description.

Important Do not use the following characters in topic, queue, user, group, server, or cluster names: period (.), asterisk (*), pound sign (#), or backslash(\).

In addition, do not use the dollar sign ($) in a username.

There are additional restrictions on values for certain properties. These are noted in the description of each property.

Table 8. Server Initialization Properties by Function

Function	*Property*	*Page*
Server Identification	BROKER_NAME	53
	BROKER_PASSWORD	54
	DEFAULT_SOCKET_TYPE	56
	PORT	62
Log Information	BROKER_LOG	53
	LOG_BLOCK_SIZE	60
	LOG_PATH	60
	LOG_QUEUE_SIZE	61
	MAX_LOG_FILE_SIZE	61
Security	DEFAULT_QOP	56
	ENABLE_QOP_SECURITY	57
	ENABLE_SECURITY	57
	TUNNELING_PROXY_HOST	67
	TUNNELING_PROXY_PORT	67
Licensing Information	CONTROL_NUMBER	55

Table 8. Server Initialization Properties by Function (*continued*)

Function	Property	Page
Database Information	DB_CONNECT	55
	DB_PASSWORD	55
	DB_PROPERTIES	55
	DB_USER	56
	JDBC_DRIVER	60
Clustering Information	ENABLE_INTERBROKER	58
	IB_CONFIG_SERVER	58
	INTERBROKER_ACCEPTOR	59
SSL Security	SSL_CA_CERTIFICATES_DIR SSL_CA_CERTIFICATES_DIR_*n*	64
	SSL_CERTIFICATE_CHAIN SSL_CERTIFICATE_CHAIN_*n*	65
	SSL_CERTIFICATE_CHAIN_FORM SSL_CERTIFICATE_CHAIN_FORM_*n*	65
	SSL_CLIENT_AUTHENTICATION SSL_CLIENT_AUTHENTICATION_*n*	66
	SSL_CIPHER_SUITES SSL_CIPHER_SUITES_*n*	65
	SSL_PRIVATE_KEY SSL_PRIVATE_KEY_*n*	67
	SSL_PRIVATE_KEY_FORM SSL_PRIVATE_KEY_FORM_*n*	67
	SSL_PRIVATE_KEY_PASSWORD SSL_PRIVATE_KEY_PASSWORD_*n*	67

Table 8. Server Initialization Properties by Function (*continued*)

Function	*Property*	*Page*
Routing Information	CONNECT_ATTEMPT_INTERVAL	54
	CONNECT_IDLE_TIMEOUT	54
	CONNECT_RETRY_COUNT	54
	CONNECT_RETRY_INTERVAL	54
	DEFAULT_ROUTING_ACCEPTOR	56
	ENABLE_LOAD_BALANCING	57
	ROUTING_NODE_NAME	63
Dead Message Queue	DMQ_OVERRIDE_MAXSIZE	57
	DMQ_NOTIFY_FACTOR	56
	ENABLE_DYNAMIC_QUEUE_CLEANUP	57
	INDOUBT_RECONNECT_INTERVAL	58
	INDOUBT_TIMEOUT	58
	ROUTING_TIMEOUT	63
	QUEUE_CLEANUP_INTERVAL	62
Point-to-Point	GUAR_QUEUE_SIZE	58
	OUTPUT_QUEUE_SIZE	62
	QUEUE_DELIVERY_THREADS	63
	WAIT_QUEUE_SIZE	68
Acceptors	IP_OR_HOST_n	59
	NUM_ACCEPTORS	61
	PORT_NUM_n	62
	SOCKET_TYPE_n	64

Table 8. Server Initialization Properties by Function (*continued*)

Function	*Property*	*Page*
Transactions	TXN_FLUSH_THREADS	68
	TXN_THREADS	68
Metrics	METRICS_COLLECTION_INTERVAL	61
	METRICS_REFRESH_INTERVAL	61

The properties given in Table 8 are described below, listed in alphabetical order. In this list, alternative values are enclosed in braces ({}) and delimited by the or symbol (|), and configurable values are shown in *italics*. Items enclosed in square brackets ([]) are optional. Default values are listed in **boldface**. No spaces are allowed before or after the equal sign (=).

BROKER_LOG

BROKER_LOG=*pathname*

This property specifies the path for the log file. The *pathname* can be relative to the server installation directory. If this line is omitted, there will be no log file, and logging information will be output to the console. See "Server Status/Error Log" on page 101 for details on the log file.

BROKER_NAME

BROKER_NAME={**SonicMQ**|*name*}

Each server in your environment must have a unique *name*, which cannot be longer than 15 characters. Leading and trailing spaces are ignored. The double quote character (") cannot be used in names. SonicMQ supports all Unicode characters, but characters must be legal for table names in your database. See "SonicMQ and Database Character Sets" on page 24.

If you edit the broker.ini file and change the BROKER_NAME, you must re-initialize the database since the table names used by the server incorporate the value of BROKER_NAME. Messages and server settings stored in the database under the old BROKER_NAME will not automatically be associated with the new server instance.

BROKER_PASSWORD

BROKER_PASSWORD=*password*

You must supply a server password for servers in a security-enabled multi-server configuration.

CONNECT_ATTEMPT_INTERVAL

CONNECT_ATTEMPT_INTERVAL=*seconds*

This property specifies the number of seconds to wait after all attempts to re-establish a connection with a routing node have failed before repeating a sequence of connection attempts. This occurs when connections are first made, when connections have timed out, or failed connections have been retried CONNECT_RETRY_COUNT times.

CONNECT_IDLE_TIMEOUT

CONNECT_IDLE_TIMEOUT={**300**|*seconds*}

This property specifies the length of time in seconds before an inactive routing connection is disconnected. This value is used if the routing connection table does not have an entry for the remote routing node. The default is 5 minutes or 300 seconds.

CONNECT_RETRY_COUNT

CONNECT_RETRY_COUNT=*number*

This property specifies the number of times that a server in a routing node will attempt to re-establish its original connection, if the connection has failed and messages remain in the routing queue. After CONNECT_RETRY_COUNT attempts, the server will ask for a new load-balanced connection.

CONNECT_RETRY_INTERVAL

CONNECT_RETRY_INTERVAL=*seconds*

This property specifies the number of seconds between retries, in the event that a server in a routing node has become disconnected from another server and is attempting to reconnect.

CONTROL_NUMBER

CONTROL_NUMBER=*value*

This property specifies the key you must have in order to run the product. The control number is the encrypted key that you supplied at installation time. It is written into the `broker.ini` file by the installation procedure. Do not change this value.

DB_CONNECT

DB_CONNECT=*database_connection*

This required property specifies the connection information for the database. For the supported databases the values of *database_connection* are:

- **jdbc:cloudscape:*pathname***, for the embedded Cloudscape database, where *pathname* is the absolute path of the database directory created when SonicMQ installs the database
- jdbc:oracle:thin:@*host*:*port*:*segment* for an Oracle 8i database, where the default *port* is 1521
- jdbc:seqlelink://*host*:*port*;databaseName=*name* for a Microsoft SQL Server 7 database

DB_PASSWORD

DB_PASSWORD=[*password*]

If the database requires a password, you supply the password through this property. The embedded database does not require a password.

DB_PROPERTIES

DB_PROPERTIES=*config_file*

The *config_file* is the name of the `cfg` file for the database together with the relative or absolute path if the `cfg` file is not in the same directory as `broker.ini`. The names of the database configuration files for supported databases are:

- **`DefaultDB_jdbc.cfg`** for the embedded Cloudscape database
- `orasvr8_jdbc.cfg` for an Oracle 8i database
- `mssqls.cfg` for a Microsoft SQL Server 7 database

DB_USER

DB_USER=[*user_name*]

If the database requires a username, you supply it through this property. The embedded database does not require a username.

DEFAULT_QOP

DEFAULT_QOP={**NONE**|INTEGRITY|PRIVACY}

Sets the default Quality of Protection (QoP) when security and QoP security are enabled and no specific QoP is set for a new or existing topic.

DEFAULT_ROUTING_ACCEPTOR

DEFAULT_ROUTING_ACCEPTOR=[{**tcp://**|ssl://}]*url*:*port*

Specifies the preferred URL for indoubt resolution connections, and optionally the protocol and port number. This property is optional. If you are using firewalls or multiple acceptors, you should set this parameter. The DEFAULT_ROUTING_ACCEPTOR comes into play when the outbound connection times out or fails, and the remote node needs to reconnect to resolve indoubt messages for the original node.

If this property is not set, the server will attempt to make a "best guess" for this URL, based on the server installation information.

DEFAULT_SOCKET_TYPE

DEFAULT_SOCKET_TYPE={tcp|http|ssl}

Indicates the protocol that the server will be using for communication.

DMQ_NOTIFY_FACTOR

DMQ_NOTIFY_FACTOR={**0.85**|*fraction*}

When a message is sent to the Dead Message Queue causing the total size of messages stored to exceed this fraction of its maximum, a dmqstatus event notification will be sent to Explorer, Admin Tool, and the administration API.

DMQ_OVERRIDE_MAXSIZE

DMQ_OVERRIDE_MAXSIZE=*kilobytes*

If you set this property, it starts the server with a new maximum size value for the Dead Message Queue. You should only set this value to allow the server to start to run clients that clear the Dead Message Queue. A server starting in this mode will display a status message to the normal output window.

If you then change the maximum size through Explorer, Admin Tool, or the administration API, the new maximum size is stored in the database and remains in effect for the remainder of the server session. Remove DMQ_OVERRIDE_MAXSIZE before restarting the server to allow the stored value to be used again.

ENABLE_DYNAMIC_QUEUE_CLEANUP

ENABLE_DYNAMIC_QUEUE_CLEANUP={TRUE|**FALSE**}

Setting this property to TRUE causes a thread to be started that does periodic checks of queues for expired messages.

ENABLE_LOADBALANCING

ENABLE_LOADBALANCING={**TRUE**|FALSE}

This property controls load balancing and is used in the management of connections between routing nodes. When this property is TRUE, the default, the server is willing to have a connection to a remote server redirected to a different server for the purpose of load balancing.

ENABLE_SECURITY

ENABLE_SECURITY={TRUE|**FALSE**}

This property controls security. It must be set to TRUE to enable security and to FALSE to disable security.

ENABLE_QOP_SECURITY

ENABLE_QOPSECURITY={**TRUE**|FALSE}

This property enables or disables the ability to set Quality of Protection (QoP) on topics. To use this, you must also enable security by setting ENABLE_SECURITY=TRUE. If QoP security is enabled and the privacy and integrity setting is used, then messages will be encrypted. Even if Secure Socket Layer (SSL) is not being used, message contents will be protected.

ENABLE_INTERBROKER

ENABLE_INTERBROKER={TRUE|**FALSE**}

This property must be set to TRUE if the server is to function as a configuration server or as a member of a cluster.

GUAR_QUEUE_SIZE

GUAR_QUEUE_SIZE={150000|*size*}

This property specifies the maximum size in bytes of server-side buffers used to maintain a list of messages waiting to be acknowledged by a receiver or subscriber to a topic or a queue.

IB_CONFIG_SERVER

IB_CONFIG_SERVER={**tcp://**|http://|ssl://}*host_name*{**:2506**|:*port_number*}

This property must be set if the server is a member of a cluster, but is not the configuration server for the cluster. In that case, the value must point to the configuration server.

INDOUBT_RECONNECT_INTERVAL

INDOUBT_RECONNECT_INTERVAL={**300**|*seconds*}

This property specifies the number of seconds that a SonicMQ server will wait between attempts to re-establish a connection with another server in order to resolve indoubt messages

INDOUBT_TIMEOUT

INDOUBT_TIMEOUT={**36000**|*seconds*}

This property specifies the number of seconds that messages are in an indoubt state—sent, but not acknowledged—before they are automatically expired.

INTERBROKER_ACCEPTOR

INTERBROKER_ACCEPTOR=*acceptor_number*

This property specifies what acceptor must be used to listen for connection requests from other servers in a server cluster. The value of this parameter must be between 1 and NUM_ACCEPTORS inclusive. It is used to determine the protocol, host name, and port number of the acceptor.

For example, if you specify INTERBROKER_ACCEPTOR=2, the protocol is set to the value of SOCKET_TYPE_2, the host name is set to the value of IP_OR_HOST_2, and the port number is set to the value of PORT_NUMBER_2. These values are used by other servers in the cluster to connect to this server.

If the INTERBROKER_ACCEPTOR property is not specified, the connection parameters are:

- **protocol** — The value of DEFAULT_SOCKET_TYPE.
- **host name** — If the NUM_ACCEPTORS parameter is specified, the value of IP_OR_HOST_1; otherwise, the IP host name or IP host address of the computer where the server is running.
- **port number** — The value of PORT.

If you do not define SOCKET_TYPE_*n*, the DEFAULT_SOCKET_TYPE protocol is used.

All servers in the cluster must use the same protocol for connections to other servers in the cluster.

IP_OR_HOST_*n*

IP_OR_HOST_*n*={*IP_address*|*host_name*}

This property specifies the machine used for acceptor *n*. It is only useful when the server resides on a machine with multiple addresses. When establishing a routing node, set the value of each IP_OR_HOST_*n* to an address of the machine where the server resides. Each IP_OR_HOST_*n* is associated with a socket type and a port number. For the first acceptor, use the IP_OR_HOST_1, DEFAULT_SOCKET_TYPE, and PORT properties. For subsequent acceptors, use IP_OR_HOST_*n*, SOCKET_TYPE_*n*, and PORT_NUMBER_*n*, where $n = 2, \ldots,$ NUM_ACCEPTORS.

JDBC_DRIVER

JDBC_DRIVER=*driver_name*

This required property tells the server the name of the JDBC driver to connect to the database. The supported JDBC drivers are:

- COM.cloudscape.core.JDBCDriver for a Cloudscape database
- oracle.jdbc.driver.OracleDriver for an Oracle 8i database using the JDBC Thin Driver
- com.merant.sequelink.jdbc.SequeLinkDriver for a Microsoft SQL Server 7 database

The SONICMQ_DBCLASS environment variable must be set in startbr.bat (Windows) or startbr.sh (UNIX or Linux) whenever JDBC_DRIVER is given a nondefault setting.

LOG_BLOCK_SIZE

LOG_BLOCK_SIZE={*num_bytes*|**8192**}

This property helps you optimize the writing of log events to the log file from memory. Disk controllers and drivers have their own limits when they write data to disk (and these limits do not necessarily match the block size of the file system). There is not much difference in performance between writing 1KB worth of log events and 8KB if the controller/driver typically writes chunks of data in blocks up to 8KB at a time. For most systems, a value of 8192 (8KB) works quite well. If, however, a controller/driver on a system writes in chunks up to 4KB, for example, resetting this property to 4096 (4KB) should improve performance. The optimal value for this property depends on the system and device.

LOG_PATH

LOG_PATH=*pathname*

This property sets the location of the log directory, the location of the server recovery logs. The *pathname* can be relative. If this property is not specified, the location will be *install-dir*/log. For more information, see "Server Status/Error Log" on page 101.

LOG_QUEUE_SIZE

LOG_QUEUE_SIZE={500000|*size*}

This property specifies the maximum size in bytes of the server-side queue of log events waiting to be written to the log. Once the total size of waiting log events exceeds this limit, threads needing to add events will block until the events currently on the queue have been written to disk.

MAX_LOG_FILE_SIZE

MAX_LOG_FILE_SIZE={**1000000**|*size*}

This property specifies the maximum size in bytes of each of the two server recovery logs. For more information, see "Server Recovery Logs" on page 102.

METRICS_COLLECTION_INTERVAL

METRICS_COLLECTION_INTERVAL={**10**|*time*}

This property specifies how far back in time metrics will be maintained. The value `time` is measured in minutes and can vary from 5 to 20.

METRICS_REFRESH_INTERVAL

METRICS_REFRESH_INTERVAL={**20**|*time*}

This property specifies the frequency with which the server will update those metric calculations that depend on a time interval and will purge expired metrics data. The value *time* is measured in seconds and can vary from 5 to 60.

NUM_ACCEPTORS

NUM_ACCEPTORS={**1**|*number*}

This property indicates the number of acceptors that are to be created and used by the server to listen for clients. There is no hard upper limit on the number of acceptors. However, limits on system resources, such as file descriptors, might dictate an upper bound.

Important Numbering for acceptors starts at 1.

You must define all acceptors prior to starting the server.

OUTPUT_QUEUE_SIZE

OUTPUT_QUEUE_SIZE={150000|*size*}

This property specifies the maximum size in bytes of the server-side output buffers used to send outgoing messages to remote clients. Setting this parameter to a larger value will allow more messages to be buffered for each client at the risk of increasing message latency.

PORT

PORT={**2506**|*port*}

This property specifies the port number used by the server to listen for incoming client requests. If PORT_NUMBER_1 is not defined, this is the port for the first acceptor.

Important If multiple servers are running on the same host, each must have a different port number. For a UNIX or Linux deployment, port numbers 1–1024 are reserved for system use, so you must pick a port number greater than 1024.

Do not attempt to start a server with a port that is being used by another application or server.

PORT_NUMBER_*n*

PORT_NUMBER_n=*port*

This property specifies the port used by an additional acceptor socket listener. The value of PORT is used for the first acceptor. Use PORT_NUMBER_*n*, where *n* is a number from 2 to NUM_ACCEPTORS, for additional acceptors. Each acceptor must have a unique port number.

Important For a UNIX or Linux deployment, port numbers 1–1024 are reserved for system use, so you must pick PORT_NUMBER_*n* greater than 1024.

Do not specify a port number that is being used by another application.

QUEUE_CLEANUP_INTERVAL

QUEUE_CLEANUP_INTERVAL={**7200**|*seconds*}

If you set ENABLE_DYNAMIC_QUEUE_CLEANUP=TRUE, you can also set this property to specify the approximate time interval between successive queue cleanups. The default is 7200 seconds, or two hours.

QUEUE_DELIVERY_THREADS

QUEUE_DELIVERY_THREADS={**1**|*num_threads*}

This property determines the number of dispatch threads that are created (and started) for dequeueing messages from queues. Increasing the number of dispatch threads might improve performance when using queues. However, as the thread count grows, a resource and lock contention might result in performance degradation. In testing with up to 300 queues, setting *num_threads* between 10 and 15 has produced better overall throughput for large numbers of queues.

ROUTING_NODE_NAME

ROUTING_NODE_NAME={**SonicMQ**|*name*}

This required property specifies the name that is used to direct messages to a queue on a particular routing node. For example, if a global queue Q1 exists on a routing node with ROUTING_NODE_NAME=node1, the queue can be addressed from a neighboring node as node1:Q1. Each server in the same routing node must have the same ROUTING_NODE_NAME. The *name* can have up to 256 characters and cannot contain double colons (**::**), a dollar sign (**$**), or double quotes (**"**). SonicMQ supports all Unicode characters, but characters must be legal for table names in your database. See "SonicMQ and Database Character Sets" on page 24.

Important The ROUTING_NODE_NAME property is required even if you are not using dynamic routing.

ROUTING_TIMEOUT

ROUTING_TIMEOUT={**7200**|*seconds*}

This property specifies the number of seconds that a message will be retained if its routing cannot be established. After this period, the message expires. Depending on message settings, the message might be transferred to the Dead Message Queue and an administrative notification might be sent. This property is server specific, and the timing starts from the first attempt to deliver the message to an adjoining routing node. The default is 7200 seconds, or two hours.

SOCKET_TYPE_*n*

SOCKET_TYPE_*n*={tcp|http|ssl}

This property specifies the protocol used by acceptor *n*, where *n* is a number from 2 to NUM_ACCEPTORS. If SOCKET_TYPE_*n* of a nondefault acceptor is not specified, it defaults to DEFAULT_SOCKET_TYPE.

For each SOCKET_TYPE_*n* entry in the broker.ini file, there must be a corresponding PORT_NUMBER_*n* entry. For example, if you set NUM_ACCEPTORS=2 in the broker.ini file, you can also set these entries:

SOCKET_TYPE_2=*protocol* (optional)
IP_OR_HOST_2={*IP_address*|*host_name*} (optional)
PORT_NUMBER_2=*port_number* (required)

SSL_CA_CERTIFICATES_DIR

SSL_CA_CERTIFICATES={**certs/ca**|*relative_path*|NONE}

This property specifies the location of the Certificate Authority (CA) certificate. The default is certs/ca. This property applies only to SonicMQ Professional Developer and E-Business Editions.

The path must be given relative to the server installation directory. For example, if SonicMQ is installed in C:\SonicMQ, the default Certificate Authority certificate would be located in C:\SonicMQ\certs\ca and the property setting would be SSL_CA_CERTIFICATES_DIR=certs/ca.

If you set the SSL_CA_CERTIFICATES_DIR property to NONE you disable the client's verification of the server's SSL security certificate chain during the connection handshake. During a normal SSL connection, the server presents a certificate chain, which the client then verifies against a list of trusted certificates. Using NONE causes verification to be bypassed.

SSL_CERTIFICATE_CHAIN

SSL_CERTIFICATE_CHAIN={**certs/serverCertChain.chain**|*pathname*}

This property specifies the pathname that points to the file containing the server certificate chain for SSL. This property applies only to SonicMQ Professional Developer and E-Business Editions. The path and filename must be given relative to the server's installation directory.

You can have certificates and private keys written directly to files, referred to as **raw**, **DER encoded,** or **binary** files. **Base64** encoding provides a convenient way to represent binary data in a textual form suitable for inclusions in, for example, mail messages. The binary data is mapped into a 64-character alphabet that includes no white space characters, so that you can add simple formatting without corrupting the data.

The binary file is the only format that is currently supported.

SSL_CERTIFICATE_CHAIN_FORM

SSL_CERTIFICATE_CHAIN_FORM={**LIST**|PKCS12|PKCS7}
SSL_CERTIFICATE_CHAIN_FORM_*n*={**LIST**|PKCS12|PKCS7}

This property specifies the format of the file containing the server certificate chain. The default is a comma-delimited list of pathnames that point to files containing each individual certificate in the chain.

Use the second form of the property for nondefault acceptors, with *n* between 2 and NUM_ACCEPTORS inclusive.

SSL_CIPHER_SUITES

SSL_CIPHER_SUITES={**null**|*list_of_cipher_suites}*

The value *list_of_cipher_suites* is a comma-delimited list of all ciphersuites supported by the server, in priority order. For server-client communications, the server determines the ciphersuite used. For server-server communications, the server that initiates the connection negotiates to the highest-priority ciphersuite they both support. This property applies only to SonicMQ Professional Developer and E-Business Editions.

The supported ciphersuites are:

- SSL_RSA_WITH_NULL_MD5
- SSL_RSA_WITH_NULL_SHA
- SSL_RSA_EXPORT_WITH_RC4_40_MD5
- SSL_RSA_WITH_RC4_MD5

- SSL_RSA_WITH_RC4_MD5
- SSL_RSA_WITH_RC4_SHA
- SSL_RSA_WITH_RC4_SHA
- SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
- SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DH_DSS_EXPORT_WITH_DES40_CBC_SHA
- SSL_DH_DSS_WITH_DES_CBC_SHA
- SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA
- SSL_DHE_DSS_WITH_DES_CBC_SHA
- SSL_DHE_DSS_WITH_DES_CBC_SHA
- SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA
- SSL_DHE_RSA_WITH_DES_CBC_SHA
- SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DH_anon_EXPORT_WITH_RC4_40_MD5
- SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA
- SSL_DH_anon_WITH_DES_CBC_SHA
- SSL_DH_anon_WITH_3DES_EDE_CBC_SHA.

SSL_CLIENT_AUTHENTICATION

SSL_CLIENT_AUTHENTICATION={TRUE|**FALSE**}
SSL_CLIENT_AUTHENTICATION_*n*={TRUE|**FALSE**}

This property specifies whether the server should enable SSL peer authentication. If this property is set to FALSE, client authentication can still be done with username and password. This property applies only to SonicMQ Professional Developer and E-Business Editions.

Use the second form of the property for nondefault acceptors, with *n* between 2 and NUM_ACCEPTORS inclusive.

SSL_PRIVATE_KEY

SSL_PRIVATE_KEY={**certs/serverKey.pkcs8**|*pathname*}
SSL_PRIVATE_KEY_*n*={**certs/serverKey.pkcs8**|*pathname*}

This property provides the pathname that points to the file containing the server-encrypted private key for SSL. This property applies only to SonicMQ Professional Developer and E-Business Editions.

The path and filename must be given relative to the server's installation directory.

Use the second form of the property for nondefault acceptors, with *n* between 2 and NUM_ACCEPTORS inclusive.

SSL_PRIVATE_KEY_FORM

SSL_PRIVATE_KEY_FORM={**PKCS8**|*format*}
SSL_PRIVATE_KEY_FORM_*n*={**PKCS8**|*format*}

This property specifies the format of the file containing the encrypted key. This property applies only to SonicMQ Professional Developer and E-Business Editions.

Use the second form of the property for nondefault acceptors, with *n* between 2 and NUM_ACCEPTORS inclusive.

SSL_PRIVATE_KEY_PASSWORD

SSL_PRIVATE_KEY_PASSWORD={**Pass Phrase**|*password*}
SSL_PRIVATE_KEY_PASSWORD_*n*={**Pass Phrase**|*password*}

This property provides the password that is used to encrypt the server private key for SSL. This property applies only to SonicMQ Professional Developer and E-Business Editions.

Use the second form of the property for nondefault acceptors, with *n* between 2 and NUM_ACCEPTORS inclusive.

TUNNELING_PROXY_HOST

TUNNELING_PROXY_HOST={*IP_address*|*hostname*}

This property specifies the IP address or hostname of the proxy server for SSL tunneling.

TUNNELING_PROXY_PORT

TUNNELING_PROXY_PORT=*port*

This property specifies the port number of the proxy server for SSL tunneling.

TXN_FLUSH_THREADS

TXN_FLUSH_THREADS={**1**|*num_threads*}

This property specifies the number of threads dedicated to flushing transactional messages to disk.

TXN_THREADS

TXN_THREADS={**10**|*num_threads*}

This property specifies the number of threads dedicated to handling transaction commit/rollback requests.

WAIT_QUEUE_SIZE

WAIT_QUEUE_SIZE={**75000**|*size*}

This property specifies the maximum size in bytes of server-side buffers for outgoing waiting messages. The wait queue buffer is used to hold messages that arrive for a subscriber or receiver while a connection is being created.

Starting and Stopping SonicMQ

Once you have initialized your database and server settings, you are ready to use SonicMQ.

Starting the SonicMQ Server

If you have a basic installation (See "Basic Installation" on page 40) the procedure to start SonicMQ is simple.

➤ **To start an application using SonicMQ (basic installation):**

1. On Windows, start the server by selecting **Start > Programs > Progress SonicMQ > Start Broker**. On UNIX, navigate to the directory containing the `startbr.sh` script and run the script.
2. If desired, start the Admin Tool or Explorer. On Windows, select **Start > Programs > Progress SonicMQ >Admin** or **Start > Programs > Progress SonicMQ > Explorer**. On UNIX or Linux, use the `admin.sh` or `explorer.sh` script.
3. Start your application.

Note You should start with the sample applications described in *Getting Started with SonicMQ*.

If you have enabled security for your server (see "Server Initialization File" on page 49), you can use the Admin tool or Explorer to set up any users, passwords, groups, topics, and Access Control Lists that might be necessary for your application. See "Security" on page 86 for more details.

If you are running a multi-server configuration on one machine, you will have installed each server in a different directory. In this case, follow the directions below.

➤ **To start multiple servers on one machine:**

1. For each server except the last installed, run `startbr.bat` (Windows) or `startbr.sh` (UNIX or Linux) from the installation directory.
2. On Windows, you have the option of starting the last server installed by selecting **Start > Programs > Progress SonicMQ > Start Broker**. On UNIX or Linux, you must start this server using `startbr.sh`.

Once you have installed and started multiple servers, you can start an administration client (Explorer or Admin) and use it to set up and administer clusters as described in "Clusters" on page 114 (Explorer) or in "Server and Cluster Administration Commands" on page 198 (Admin).

Stopping the SonicMQ Server

To safely shut down the SonicMQ server, you must use one of the administrative tools:

- To shut down a SonicMQ server using Explorer, use the **Shutdown** button, as described in "Connecting to a Server" on page 110.
- To shut down a SonicMQ server using Admin Tool, use the `shutdown broker` command, as described in "shutdown broker" on page 200.

Note You can also shut down the server programmatically. See the "Management API" chapter of the *SonicMQ Programming Guide* for an example.

Setting Up SonicMQ as a Windows Service

On a Windows server, you can set up a Sonic MQ server to run as a Windows service. Once you have set up a server as a Windows service, it can be run automatically or manually launched and shut down. If run automatically, it will launch at system startup and shut down at system shutdown.

Before installing a server to run as a Windows service, you must have the proper administrative rights, as explained in the following section.

Windows Administrative Rights

To install a Windows service you must have Administrator rights on the Windows server machine.

On Windows NT only you must also have the **Log on as a service** right. Because this right is not active by default, you must give yourself the **Log on as a service** right if you do not already have that right.

➤ **To assure that you have the Log on as a service right (Windows NT only):**

1. From the Windows **Administrator Tools** applications group, start the **User Manager** application.
2. On the **Policies** menu, choose **User Rights** to display the **User Rights Policy** dialog box.
3. Set the check box for **Show Advanced User Rights** to display all the user rights in the **Right** drop-down list box.
4. Click on the **Right** drop-down list box and select the **Log on as a service** item from the displayed choices:
   - If the group you belong to remains displayed in the **Grant to** list box, then you already have the required right. Skip to Step 9.
   - If you do not have the **Log on as a service** right, you can grant that right to yourself by performing Step 5 through Step 9.
5. Choose **Add**. The **Add Users and Groups** dialog box appears.
6. Select the your username or your group name to grant yourself the **Log on as a service** right.

7. After you have added yourself, choose **OK** to close the **Add Users and Groups** dialog box.
8. In the **User Rights** dialog box, your username or your group name is displayed in the **Grant to** list box. Choose **OK** to close this dialog box.
9. Close the **User Manager** application.

Installing and Removing SonicMQ as a Windows Service

Once you have the proper administrative permissions, you can install or remove a SonicMQ server as a Windows service. For a multi-server configuration, you must install each server separately.

➤ To install a server as a Windows service:

☐ In a DOS window, enter the command:

```
SonicServiceSetup -install service_name [-a]
-jre=jre_path\jre.exe -home=homepath\broker.ini
[-b=serverurl] [-u=serveruser] [-p=serverpass] [-cp=classpath]
[-inipwd=passwd] [-encpwd=encrypted-passwd] any_java_parameters
```

where the parameters are explained in Table 9.

➤ To remove the SonicMQ service:

1. If the service is running, stop the service.
2. In a DOS window, enter the command:

```
SonicServiceSetup -remove service_name
```

where the parameters are explained in Table 9.

Table 9. SonicServiceSetup Parameters

Parameter	*Description*
-a	Starts automatically when the Windows machine is started. If this parameter is omitted, the service is installed in manual mode. You can start it using the **Services** tool in the **Control Panel**.
-b=*serverurl*	Specifies the connection URL to the server in the form {**tcp://**|http://|ssl://}hostname{**:2506**|:port} The default is localhost:2506.
-encpwd=*encrypted-passwd*	If you are using an encrypted server initialization file with an encrypted password (produced by PBETool), specifies the encrypted password needed to decrypt it. See the "Security" chapter of the *SonicMQ Deployment Guide*.
-home=*homepath**file_name*	Specifies the path of the server initialization file and the name of the file used to start the server. *file_name* is broker.ini unless you have renamed the file.
-inipwd=*passwd*	If you are using an encrypted server initialization file (produced by PBETool), specifies the clear-text password needed to decrypt it. See the "Security" chapter of the *SonicMQ Deployment Guide* for details.
-install *service_name*	Installs *service_name* as a Windows service, where service_name is the name you choose to give to the service.
-jre=*jre_path*\jre.exe	Specifies the path to the Java executable used by the server.
-p=*serverpass*	Specifies the password used to connect with the server. The default is Administrator.
-remove *service_name*	Removes service_name as a Windows service.
-u=*serveruser*	Specifies the username used to connect with the server. The default is Administrator.
-cp=*classpath*	For the IBM Version 1.1.8 JVM, this parameter is required. For the Sun Version 1.2.2 JVM, the parameter is not required if you have set the system level CLASSPATH environment variable to the classpath required to run the service. Obtain the value of this parameter by copying the classpath in startbr.bat.
any_java_parameters	Specifies arguments for the JVM such as heap size.

Running SonicMQ as a Windows Service

You run SonicMQ differently depending on whether your operating system is Windows NT or Windows 2000.

➤ **To run SonicMQ automatically as a Windows NT service:**

1. From the **Control Panel**, double-click the **Services** icon.
2. Select the name you have given to the SonicMQ service and click the **Startup** button.
3. Set **Startup Type** to **Automatic**.

➤ **To run SonicMQ automatically as a Windows 2000 service:**

1. From the **Control Panel**, double-click the **Administrative Tools** icon.
2. From **Administrative Tools**, double-click the **Services** icon.
3. Right-click on the name you have given to the SonicMQ service and set **Startup Type** to **Automatic**.

Troubleshooting a SonicMQ Windows Service

If the SonicMQ Windows service does not start, examine the file `SonicService.log` in the SonicMQ installation directory. Make sure that you have correctly specified the classpath and executable.

Testing the Server

You can test your server by running one of the sample applications that are packaged with SonicMQ and described in *Getting Started with SonicMQ*.

You can run the samples using a multi-server configuration by including the parameter `-b hostname:port`, where `port` is the port number of any server in the cluster and `hostname` can be `localhost` if the server is on the same machine as the client.

An even easier way to test your server is by using the SonicMQ Explorer to:

- Create a session
- Publish to a topic
- Subscribe to a topic
- Send a message from a publisher to one or more subscribers

See "Sessions and Connections" on page 150 for details.

Troubleshooting the Server Database

The file *install-dir*`\process_install.log` is created upon installation and contains a record of errors that might have occurred in configuring the default database. Examine this file if you have trouble with the default server database.

The possible sources of trouble with the default server database are:

- The server cannot find the database.
- The database is not properly configured.

If the server cannot find the default database, you must edit the `broker.ini` file to tell the server the location of the database.

➤ **To edit the server initialization file for the database location:**

1. In your preferred editor, open the `broker.ini` file in the SonicMQ installation directory.
2. Make sure that there is exactly one uncommented line beginning with `DB_CONNECT`. If you are using the default database it should read `DB_CONNECT=jdbc:cloudscape:`*`database_path`*. If you are using a nondefault supported database, see "DB_CONNECT" on page 55 for the proper value.
3. Change *`database_path`* to the absolute path to the database.
4. Save the `broker.ini` file.

If the database is not properly configured or has become corrupted, you will have to reinitialize it.

➤ **To reinstall the server database:**

1. In a Command Prompt window, locate the prompt in the SonicMQ installation directory.
2. Re-create the basic database tables, as described in Table 7 on page 48.

The server database performs initialization and provides comments as it proceeds.

Installing Only the SonicMQ Client

The standard SonicMQ installation installs the client and the message server. If a machine requires only the client (runs only a client that accesses a message server on the network), you can install only the client on that machine.

To install only the SonicMQ files used by a client, the recommended procedure is to access an existing complete SonicMQ installation and copy selected files to the target client-only system.

Note **Java Virtual Machine —** You probably already have an appropriate JVM for the Java client that you are running. A complete SonicMQ installation installs an appropriate JVM for its platform. However, if you are creating a client-only installation on a platform with no appropriate JVM installed, you must get a JVM for the client-only platform.

See `http://www.sonicmq.com/product_info/platforms.htm` for a list of the recommended JVMs for each supported platform.

➤ To copy SonicMQ client files from a complete SonicMQ installation:

1. On the system where you want the client installation, create a new folder for the SonicMQ files. For example, `D:\SonicMQ_Client`.
2. From an accessible system that has the appropriate version of SonicMQ, copy the folder **`Java`** to the target install directory.
3. Create a new directory in the target directory, `lib`.
4. From the source system installation `/lib` directory, copy the Java archives in Table 10 to the target system's installation `/lib` directory.

Table 10. Java Archives in the Client Installation /lib

Java Archive	*Need to copy*
`webclient.jar` or `client.jar`	One is required. Use `webclient.jar` if you want to allow HTTP tunneling.
`jndi.jar`	Optional. Required if you are using ConnectionFactories to create connections.
`sonicSSL.jar`	Optional. Required if you need to use SSL and you are using `webclient.jar`, not `client.jar`.
`gnu-reg-exp-1.0.6.jar`	Optional. Used by message selector.
`xml4j.jar`	Optional. Used by XML message type.

5. Add the required SonicMQ.jar files, and any other SonicMQ.jar files that you need for your application, to your client class path; specify all the SonicMQ files on the class path together.

When you are done, the JMS functions required by the Java client perform as expected.

Important While the Admin tool and the SonicMQ Explorer are client functions, these tools require server functionality. If you want to use the **Admin** tool or the SonicMQ **Explorer**, you must do so on a system that has a complete SonicMQ installation.

Uninstalling SonicMQ

You uninstall SonicMQ differently, depending on the operating system you are using.

➤ **To uninstall servers on Windows:**

1. Choose **Start > Programs > Progress SonicMQ > Uninstall**.
2. If you have multiple servers installed on one machine, the previous step uninstalls the last-installed server. Uninstall any remaining servers by running `uninstall.bat` from each server installation directory.

➤ **To uninstall servers on UNIX or Linux:**

- ☐ For each server, go to the server installation directory and remove the directory and all subdirectories with the command `rm -r`.

Note You do not need an uninstall script to uninstall SonicMQ on a UNIX or Linux machine.

Chapter 2 Administration Concepts

In this chapter you will learn the concepts that underlie administration of a SonicMQ system. The following two chapters are devoted to explaining the use of two of the tools that SonicMQ supplies to perform the administration functions. This chapter contains the following sections:

- "Message Server Administrative Tools" on page 80 provides a high-level overview of the administration tools.
- "Clustered Servers" on page 83 discusses the basic concepts of creating multi-server (clustered) server configurations, which give SonicMQ the ability to scale.
- "Security" on page 86 explains the concepts that underlie SonicMQ security including encryption, identification and authentication, and access control.
- "Dynamic Routing Architecture" on page 99 covers the concepts of Dynamic Routing Architecture that are useful in the creation of large-scale business-to-business applications, and points you to other SonicMQ documentation for further details.
- "Logging" on page 101 describes two kinds of logs maintained by SonicMQ.
- "JMS Administered Objects" on page 102 shows how to create and maintain JMS administered objects.
- "Certificate Management" on page 104 explains the concepts of managing security certificates.

Message Server Administrative Tools

SonicMQ provides you with three tools for administration:

- Explorer, a graphical-user-interface tool, described in Chapter 3, "Graphical Administration Tool"
- Admin, a command-line tool, described in Chapter 4, "Command-line Administration Tool"
- Management API, described in the *SonicMQ Programming Guide*

Basic Functions

With each tool you can:

- Connect to, or disconnect from, a running server
- Shut down a server
- Manage security (where enabled)
- Manage the Dynamic Routing Architecture
- Manage queues and topics
- Receive and examine server events
- Examine server performance metrics
- Examine and purge a server log (where enabled)
- Establish and manage a JMS administered object store
- Manage security certificates (Explorer only)

Case Sensitivity and Character Restrictions

In SonicMQ, all names are case sensitive, including server names, user names, passwords, cluster names, and routing node names. For example, you could have two distinct users named **Administrator** and **administrator**. SonicMQ names consist of Unicode characters, and must only include characters supported by your database. See "SonicMQ and Database Character Sets" on page 24 for details. Additional restrictions on property names will be mentioned as the properties are introduced.

Cluster Management Functions

If one of the servers is set up as a configuration server so that you have the potential to establish a server cluster, you can manage the cluster from either Admin or Explorer.

You can:

- View candidates for cluster membership
- Create a cluster
- Add servers to a cluster
- Delete servers from a cluster
- Examine a cluster configuration
- Delete a cluster

The concepts that underlie clusters in SonicMQ are described in the "Clustered Servers" section on page 83.

Adding Servers to Security-enabled Clusters

Servers to be added to a security-enabled cluster must be configured as described in "Configuring Servers for a Security-enabled Cluster" on page 89.

When adding a server to a security-enabled cluster, you must follow a sequence of steps in the proper order. If you do not, you might not be able to add the server to the cluster.

➤ **To add a server to a security-enabled cluster:**

1. Start the configuration server.
2. Start an administrative client: Admin tool or Explorer.

   The next three steps are performed using the administrative client.
3. Connect to the configuration server.
4. Add the server name and server password of the intended cluster participant to the configuration server's list of users.
5. Add the new user (server) to the Administrators group.
6. Start the server that you want to add to the cluster.

7. Create the cluster.
8. Add the server to the cluster.

Security Functions

If security is enabled, you can use Admin or Explorer to manage many aspects of security, including:

- Users
- Groups
- Topic Quality of Protection
- Queue Quality of Protection
- Access Control Lists

The security concepts that underlie SonicMQ are described in the "Security" section on page 86.

JMS Session Management Functions

Sessions are established to serialize communication between a client and a server. For details, see Sun's JMS specification, *Java Messaging Service, Version 1.0.2* available from `http://java.sun.com/products/jms/docs.html`.

Admin and Explorer create default administration messaging sessions with the server. In addition, Explorer lets you create extra JMS Sessions to test server functionality by simulating your own JMS clients. This capability is not available with Admin.

Clustered Servers

Clustering is an important feature of SonicMQ Professional Developer and E-Business Editions, which enable them to offer the advantages of distributed computing, such as scalability and fault tolerance.

Overview of Clustering

Briefly stated, a **cluster** is a collection of interconnected servers. Each server within the cluster communicates directly with every other server in the cluster. A cluster is managed by a special kind of server, called a **configuration server**. Using the configuration server, you can:

- Add servers to the cluster so that each cluster member can accept connections from any another.
- Centrally administer a security database including users, groups, access control, and message protection.
- Remove servers from the cluster.

The configuration server for a cluster might or might not be part of the cluster. A configuration server can manage only one cluster.

Important If a message server is a member of a cluster, the message server waits for the configuration server to come online before completing its own startup. One result is that the SonicMQ Explorer or Admin tool cannot access the server during this startup sequence, so you cannot use the Explorer or Admin to perform administrative tasks for that server, such as connect to it or shut it down. The server is not hung; it is waiting for the configuration server to start up. If you want to shut down the server during this waiting period, you must terminate the server by pressing **Ctrl + C** in the **Start Broker** console window or by terminating the JVM process for the server.

Figure 5 shows a cluster consisting of a configuration server and three (message) servers. Clients can connect directly to any of the three servers. They can also connect to the configuration server if it is designated as part of the cluster. Effectively, the cluster appears to clients as if it were a single message server. Thus, a message that is published on one server can be received by clients connected to any other server in the cluster.

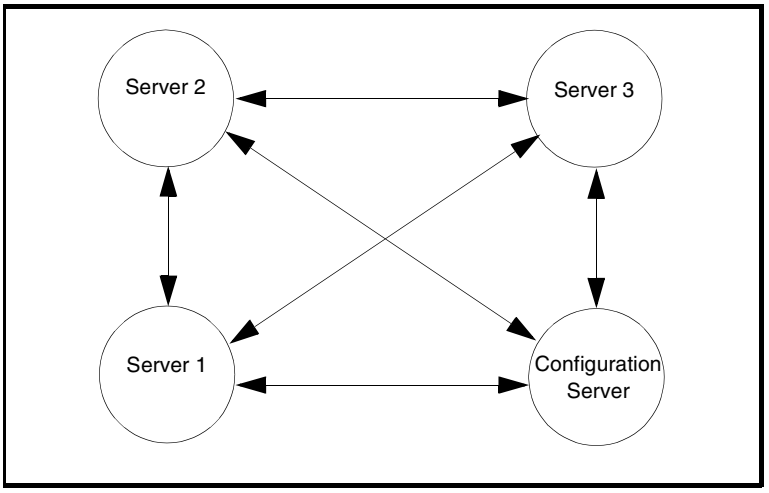


Figure 5. A Sample Cluster Configuration

Clustering and Security

Clusters allow for centralized maintenance of security functions: the management of users, groups, topic/queue Quality of Protections, and Access Control Lists. (See "Security" on page 86).

If a message server is running as part of a cluster and the server has security enabled, security functions will be managed by the configuration server and not by the message server itself.

Clustering and Fault Tolerance

By allowing a group of servers on different machines to act as a single virtual server, a cluster provides fault tolerance. Suppose an application client that has been connected to a server (say, **Server 1**) loses its connection (perhaps because its host goes down). If the client knows the address of any other server in the cluster (say, **Server 2**), it can connect to that server, where it can pick up any messages on topics to which it is subscribed. If the failure is just one of connectivity between the application client and **Server 1**, the messaging services continue normally. If **Server 1** is not operating, the client can still get all messages from the other servers.

If the configuration server goes down, all servers in the cluster still running will continue to work and they will be able to continue accepting new connections. There are only three things that will not happen while the configuration server is down:

- Servers that are not already running can be started, but they will not accept connections until they are able to connect to the configuration server after it comes back up.
- You cannot administer the security database since it is maintained by the configuration server. However, the other servers will continue to enforce security according to the state of the database at the time the configuration server went down.
- Servers cannot be added to or removed from the cluster.

Security

SonicMQ provides comprehensive security in many ways:

- Encryption and integrity at the connection level. This occurs when SSL is used, as described in "Ciphersuites and Certificates" on page 30 (Professional Developer Edition and E-Business Edition only). SonicMQ makes it convenient for you to manage your SSL security certificates. See "Certificate Management Tool" on page 177.
- Encryption and integrity at the message level, as described in "Quality of Protection" on page 87.
- Identification and authentication, described in "Identification and Authentication" on page 90. See also the information on certificate-based mutual authentication in the "Security" chapter of the *SonicMQ Deployment Guide*.
- Access control, described in "Access Control" on page 92 and the following sections.
- Firewall support, described in the "Security" chapter of the *SonicMQ Deployment Guide*.
- Password-based encryption of broker configuration files. See the description of PBETool in the "Security" chapter of the *SonicMQ Deployment Guide* for details.
- HTTP security features including HTTP Tunneling, support for forward and reverse proxies, and support for signed applets. See the "Security" chapter of the *SonicMQ Deployment Guide*.

SonicMQ security is established by defining users, groups, Quality of Protection (QoP) settings, and Access Control Lists (ACL). You can use Explorer or Admin to administer these items.

When security is enabled, clients (including administration clients) must provide a valid username for identification and a password for authentication.

SonicMQ provides data protection through a set of Quality of Protection options. These options allow the system administrator to specify whether a message will have no protection, integrity, or privacy and integrity. Quality of Protection options are based on topics (in the Publish and Subscribe domain)

or on queues (in the Point-to-Point domain). See *Getting Started with SonicMQ* for a discussion of Publish and Subscribe, Point-to-Point, topics, and queues.

In the Publish and Subscribe (Pub/Sub) domain, SonicMQ provides the ability to control who can publish or subscribe to a particular topic. The system administrator establishes these permissions by defining topic-based ACLs. See "Security on Topics" on page 91 for details. In the Point-to-Point (PTP) domain, SonicMQ provides the ability to control who can send or receive to a queue. See "Security on Queues" on page 92 for details.

Quality of Protection

Message protection takes place when a Quality of Protection (QoP) value of **integrity** or **privacy and integrity** is specified for a message topic or queue. Integrity ensures that the message body is not altered in transit, whereas privacy additionally ensures that the message body cannot be intercepted and read while in transit. Since SonicMQ always ensures that a message that has privacy also has integrity, SonicMQ sometimes uses the term **privacy** as a shorthand for privacy and integrity.

Important Quality of Protection is applied to the message body, but not to the message headers or properties.

The system administrator sets Quality of Protection policies on topics or queues. These policies can be set for an individual topic or queue, or for all topics or queues that match a template. The policy can be none, integrity, or privacy.

Integrity

Integrity verifies that the message content upon delivery matches its original published form. Corruption of data can be accidental or intentional. Communications programs commonly use a checksum algorithm to check transmitted data for accidental corruption, such as communication errors. Specialized cryptographic checksums, called message digests, can check transmitted data for intentional corruption as well. To validate the integrity of message content, SonicMQ uses the cryptographic checksum Message Digest

5 (MD5) algorithm along with a secret key to provide a Message Authentication Code (MAC). MACs are equivalent to encrypted digests and are used to protect the integrity of a message.

Privacy

SonicMQ gives a message privacy by using encryption. Encryption scrambles the message content before sending it over the wire and restores the original form upon delivery. If the message is intercepted before delivery (that is, someone attempts to read it as it goes over the wire) the data is not in readable form.

QoP and SSL

When **Secure Sockets Layer (SSL)** (Professional Developer and E-Business Editions only) is used, a message is encrypted and decrypted at the connection level. A message is encrypted just before it is sent over the wire, and it is decrypted by the connection level software at the other end of the wire. As far as the actual application is concerned, it is dealing only with unencrypted data — all encryption and decryption takes place at a lower level.

A special type of socket (on both ends of the connection) is required to support SSL.

When only an insecure connection is available, such as a plain TCP/IP or HTTP connection, you might still have to ensure the security of your communication. To do this, you can have messages encrypted and decrypted by the SonicMQ client at run time and the message server by setting a QoP value of privacy on a destination. You can set the QoP not only on a single topic or queue, but also on the collection of all topics or queues whose names match a template pattern that you select.

Note that SSL works in a connection-point to connection-point fashion, for example from a sender to a server. In contrast, privacy works in end-to-end fashion, for example from a sender to a server. If SSL is used, a message could be read if it was intercepted after it reached the server. If privacy is used it could not be read. If you use privacy, messages stored in the recovery log or database are stored on disk in encrypted form. If you use SSL, the message are stored in unencrypted form. Using QoP privacy can also be more efficient than using SSL for two reasons:

- A message need only be encrypted once to reach many subscribers.
- Routine messages that do not require protection need not be encrypted. Since encryption and decryption have performance impacts, system performance can often be significantly improved by avoiding encryption where it is not needed.

Setting Up Security

You configure a stand-alone server for security by:

- Setting `ENABLE_SECURITY=TRUE` in the `broker.ini` file
- Running `dbtool` as described in Table 7 on page 48 to initialize the security database

Configuring Servers for a Security-enabled Cluster

If you have a clustered server configuration, security is managed centrally by the configuration server. For a security-enabled cluster, you must modify the `broker.ini` file for each cluster member as follows:

- Set `BROKER_NAME` to a value that is unique within the cluster, and also different from the `BROKER_NAME` of the configuration server (which need not be in the cluster.)
- Set `BROKER_PASSWORD` to any string. This string is used in combination with `BROKER_NAME` by the configuration server to authenticate the cluster member when it tries to connect to the configuration server, and to associate the server with a user in the configuration server database.
- Set `IB_CONFIG_SERVER` to the host name and port where the configuration server will run. If you do not set the port, it defaults to 2506. Do not set `IB_CONFIG_SERVER` if the cluster member is also the configuration server.

You must also make the following settings in the `broker.ini` file for the configuration server **and** each cluster member:

- `ENABLE_INTERBROKER=TRUE`
- `ENABLE_SECURITY=TRUE`

All the above settings must be made prior to initializing the server databases with `dbtool` as described in Table 7 on page 48.

You can set many security options using Admin or Explorer. Admin or Explorer changes are effective immediately. It is not necessary to restart an application or a server to begin enforcing these new security options. For example, if a user is subscribed to a topic and the administrator denies them access to the topic, they immediately can no longer retrieve messages from the topic.

Identification and Authentication

To determine whether a user is entitled to publish/subscribe or send/receive a message, SonicMQ must first know who the user is (identification) and whether they really are who they say they are (authentication).

The default implementation for identification and authentication relies on an established username and password combination. The system administrator creates the username and password with Admin or Explorer. The administrator can grant or deny permissions for various actions to individual users or to groups of users. See "Groups and Security" on page 98 for a description of how SonicMQ resolves conflicts between user and group security permissions.

A username or group name consists of a string of Unicode characters, with the exception of the characters dot (**.**), asterisk (*****), pound (**#**), and backslash (****). The special group name **PUBLIC** is reserved by SonicMQ and should not be manually modified or deleted. For further details on **PUBLIC**, see "Access Control" on page 92.

A password consists of a string of Unicode characters. SonicMQ provides password security by **not** sending the user's password across the network, thus preventing a situation where a hostile eavesdropper could capture confidential information for their own use. This is accomplished by means of a challenge/response protocol where the server sends a challenge (based on the user's password) back to the client, which must respond successfully to be granted access. The only exception to this rule occurs when a user changes a password. In that case the password is encrypted and sent to the SonicMQ security service.

Admin or Explorer changes are effective immediately. If a user is connected to a security-enabled server and they are removed from the list of authenticated users, they will lose their connection.

Security on Topics

In the SonicMQ Publish and Subscribe domain all information flow is dependent on topics:

- Developers organize information based on topics.
- Applications register their interest in consuming information by subscribing to a topic.
- Applications produce information by publishing messages to topics.
- The SonicMQ message server routes information from publishers to subscribers based on the topic.

The security subsystem takes advantage of the information flow's dependency on topics. By protecting the topic, you can precisely and dynamically control who has access to the information. SonicMQ refers to this as topic-based security. SonicMQ associates a security policy with every topic. The policy determines the following:

- Who can publish on or subscribe to a topic?
- Do messages on a topic need to be privacy-protected (encrypted)?
- Do messages on a topic need to be integrity-protected (encrypted checksum)?

Since topics themselves are organized in a tree, the security policy of a topic can be inherited by some or all its descendants. SonicMQ enforces the security policy automatically and without any application intervention. See "Inheritance of Security Policies" on page 95 for details.

Permission conflicts (for example, when a user is denied the right to publish to a topic but the user belongs to a group that is granted access to the same topic) are settled by the rules given in "Groups and Security" on page 98.

Security on Queues

In the SonicMQ Point-to-Point domain, all information flow is through queues. Queue names can be given the same type of hierarchical structure as topic names, so that queue security policies can be enforced in exactly the same way as topic security policies.

Access Control

SonicMQ manages access control with an Access Control List (ACL). In SonicMQ, the term **principal** refers to an individual user or group. The system administrator uses a topic's ACL to give a principal permission to publish or subscribe to a topic. The administrator uses a queue's ACL to give a principal permission to send to or receive messages from the queue.

Progress delivers SonicMQ with a special group principal **PUBLIC** already established. **PUBLIC** represents all authenticated users in the system and by default, members in this group are given **all** permissions on **all** topics. To change this default behavior, you can remove the ACL entry for **PUBLIC** from the root topic `$SonicMQ-Root-Subject`.

Important

If you remove the ACL entry for **PUBLIC**, no one in the system will have **any** permission unless it is granted explicitly. For this reason, you might prefer to set **PUBLIC**'s permission explicitly for individual topics without removing it from the root topic.

Note

PUBLIC is useful for creating ACLs that give all authenticated users permission to perform an operation on a topic or queue.

When you install the SonicMQ server, the additional principals shown in Table 11 are already established.

Table 11. Special Principals

Principal Name	*Type*	*Description*
Administrator	User	Initially established with all permissions on existing topics. The default password is `Administrator`.
Administrators	Group	Initially established with all permissions on existing topics, and containing the principal user `Administrator`. Only members of this group can use the Admin or Explorer tools to perform administration tasks. This group contains all the users defined with the same permissions as `Administrator`.
Broker name	User	Created if you enable security. *Broker name* is the value of the parameter `BROKER_NAME` in `broker.ini`. The default password is `SonicMQ`. You establish a nondefault password by setting `BROKER_PASSWORD` in `broker.ini`.

The `Administrator` principal can be modified or deleted.

Warning Do not delete the `Administrator` principal unless you have created another user who is a member of the `Administrators` group. Since only members of the `Administrators` group can perform administration tasks, this would disable the administration of your SonicMQ system

For security purposes, it is important that the system administrator change the password for `Administrator` as soon as possible.

Warning Do not delete the `Administrators` group. Since only members of this group can perform administration tasks, deleting this group disables the administration of your SonicMQ system.

Access Mediation

When the SonicMQ server checks if the client is authorized, it checks not only the user's permissions but, when necessary, the permissions of the groups to which the user belongs. See "Groups and Security" on page 98 for details.

Access Mediation in the Publish and Subscribe Domain

In the Publish and Subscribe domain the server performs access mediation on a client's subscribe or publish and guaranteed delivery requests. For example, when the client subscribes to a topic, the server retrieves the policy for the topic and checks if the client is permitted to subscribe to it. If a check fails, the subscribe request is rejected and the client throws an exception at run time.

When a client publishes a message on a topic, the server checks if the client is authorized to publish on that topic. If the client is not authorized, the publish request is rejected, and the client throws an exception run time. If the client is authorized, the server delivers the message to all clients that are subscribed to the topic.

Access Mediation in the Point-to-Point Domain

Similarly, in the Point-to-Point domain, the server performs access mediation on a client's send and receive requests.

When a client sends a message to a queue, the server checks if the client is authorized to send a message to the queue. If the client is not authorized, the send request is rejected, and the client throws an exception at run time. If the client is authorized, the client sends the message to the queue.

When a client attempts to receive a message from a queue, the server checks if the client is authorized. It the client is not authorized, the send request is rejected, and the client throws an exception at run time. If the client is authorized, the client receives the message, which is then removed from the queue.

Inheritance of Security Policies

Since you can organize topics and queues in a hierarchical tree structure, the security policy of a parent topic or queue can be inherited by some or all of its descendents that do not have an explicit policy. Therefore, it is not necessary to have an explicit security policy associated with every topic or queue.

Important Since both types of destinations—topics and queues—support the same methods for handling security policies, it is not necessary to talk about them separately. For a concrete example, the following discussion is based on topics.

Every topic has an implicit security policy, which is that of its parent. As an example, consider the topic tree shown in Figure 6.

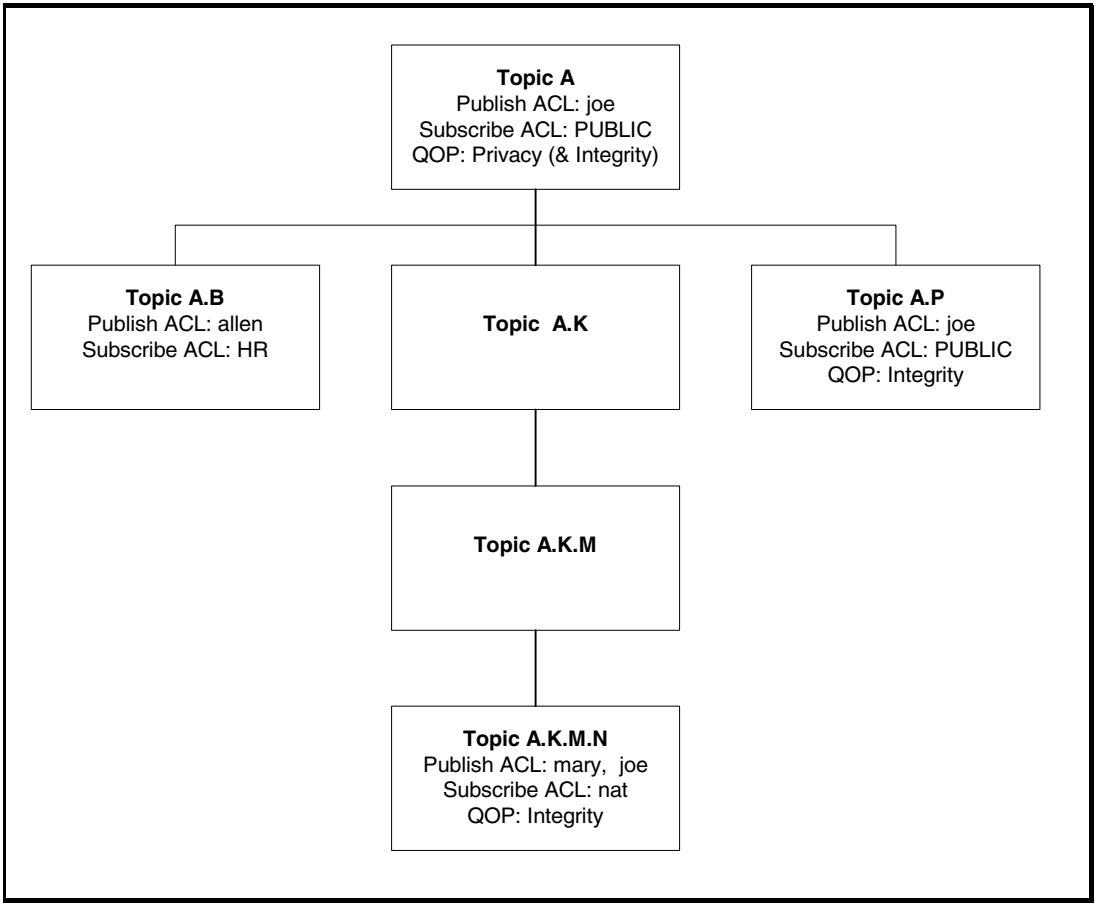


Figure 6. Topic Hierarchy with Security Policies

The Access Control List and Quality of Protection for each topic in the tree are summarized in Table 12.

Table 12. Access Control and Quality of Protection for Topics in a Hierarchy

Topic	Publishers	Subscribers	Quality of Protection		Comments
			Privacy	Integrity	
A	joe	PUBLIC	Yes	Yes	Explicit policy
A.P	joe	PUBLIC	No	Yes	Explicit policy
A.K	joe	PUBLIC	Yes	Yes	Policy inherited from A
A.K.M	joe	PUBLIC	Yes	Yes	Policy inherited from A.K
A.K.M.N	mary, joe	nat	No	Yes	Explicit policy
A.B	allen	HR	Yes	Yes	Explicit policy

Security Policies and Name Spaces

A name space is a set of topic or queue names using template characters asterisk (`*`) or pound (`#`). For example, **`A.*`** indicates any name of the form **`A.`*`X`***, where ***`X`*** is a string which does not include a dot (`.`) and **`A.#`** indicates any name of the form **`A.`*`Y`*** where ***`Y`*** is a string which may contain dots. SonicMQ lets you associate a security policy with a name space but you can override this assignment. At run time, SonicMQ correctly assigns the security policies.

For example, for the topic tree in Figure 6 on page 96, assume there is a security policy associated with **A.*,** which has an ACL entry for granting the principal *lisa* permission to *subscribe*. *lisa* will then have permission to subscribe to topics **A.B**, **A.K**, and **A.P** unless there is an ACL entry associated with any of these topics explicitly denying *lisa* that permission.

When a message is published on **A.P** or **A.K**, the message server delivers it to the user who subscribed to **A.***. However, when a message is published to **A.B**, that message is delivered only to subscribers who are in the HR group. Moreover, if the system administrator changes the subscribe ACL of any topic that matches **A.***, the server will correctly enforce the ACL at the time of message delivery. Effectively, subscribing to a wildcard topic has the semantics to deliver messages on all topics that match the template characters and for which the subscriber has authorization to receive.

Groups and Security

SonicMQ's method of assigning permissions to both users and groups of users is quite powerful. However, to use it fully, you need to understand how SonicMQ mediates apparent conflicts in permissions. Such a conflict would occur, for example, if user Joe belongs to the Sales group and the Marketing group, and Sales has publish permission to the Financials topic while Marketing has been denied permission, as illustrated in Figure 7. Does Joe have permission to publish to Financials?

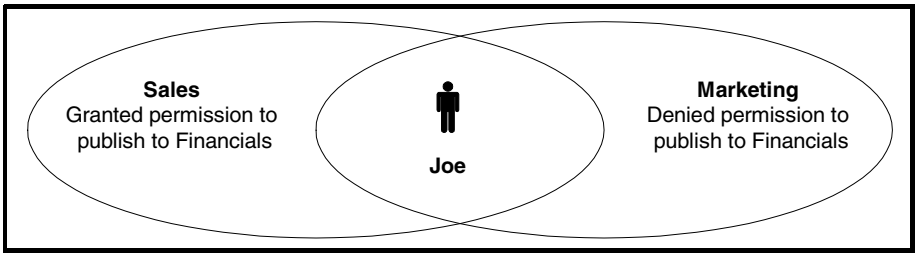


Figure 7. A Permissions Conflict

The conflict resolution rules are identical for publishing, subscribing, sending, or receiving.

Groups and users have three types of permissions for publishing, subscribing, sending, or receiving: **Grant(+)**, **Deny(-)**, and **Inherit**. The determination of whether a user has a permission depends on two rules:

1. If the user has either **Grant** or **Deny** permission, that permission overrides the permission of any groups to which the user belongs.
2. If the user has **Inherit** permission, then:
   - The user is granted permission if at least one group to which the user belongs has **Grant** permission.
   - The user is denied permission otherwise.

Thus, the question about Joe has an answer that depends on his individual publish permission:

- If Joe has **Grant** permission to publish, he will be granted permission to publish to Financials.
- If Joe has **Deny** permission to publish, he will be denied permission to publish to Financials.
- If Joe has **Inherit** permission to publish and Sales has **Grant** publish permission, then Joe has permission to publish. The fact that Marketing has **Deny** permission to publish is irrelevant.

Dynamic Routing Architecture

Dynamic Routing Architecture (DRA) is a feature that enables SonicMQ to support a multi-node environment. One type of multi-node environment is a Marketplace application that might involve thousands of servers belonging to remote Trading Partners. In this type of application a hub and spoke architecture enables the Trading Partners to communicate with each other through multiple servers in a central Portal. For more details on this application and a discussion of other architectures using DRA, see the *SonicMQ Deployment Guide*.

Note Dynamic Routing Architecture is available for the Point-to-Point domain only.

The administrative tools let you manage many aspects of the Dynamic Routing Architecture. Therefore, a few of the concepts of queue routing that you need to understand to use the administrative tools are mentioned here. For more details about these concepts, see the *SonicMQ Deployment Guide*.

DRA enables an application to send messages to a queue on a server to which it is not a client.

Routing Nodes and Global Queues

A **routing node** is either a single unclustered server or a cluster of servers. Servers in the node are identified by a **routing node name**. All servers in the same cluster must share the same routing node name. Queues that can be accessed from remote routing nodes are called **global** queues. A queue that can only be reached from direct clients of the server is called a **local** queue.

Setting a Routing

The administrative tools let you set routing information for a routing node. This information is called a **routing**. A routing contains the following routing information for the node:

- A list of connection URLs. This list typically points to a subset of the servers in the routing node. When a routing connection to a routing node is required, a connection will be made using one of these connection URLs. When a routing connection is active (running), the active connection is used and the connection URL list is not examined.
- A username associated with routing. You can change the password corresponding to this username.
- A load balance flag, which indicates whether the node will allow connection URLs to servers in the node to be reset for the purposes of load balancing.
- A list access flag, which specifies whether selection from the list of connection URLs to servers in the node should be made sequentially or randomly.
- A connection timeout, which specifies the length of time in seconds before an inactive routing connection is disconnected. A value of 0 (zero) represents an infinite timeout.
- An **advertising** flag. If advertising is turned on, information about all known global queues in the node is sent to the the routing destination node for dynamic routing configuration purposes. If advertising is turned off, the information is not sent.

- A connection flag, which specifies whether the routing is to be **static** or **dynamic**. For a static routing connection, a remote server connecting with the routing node always establishes a connection to a server that you specify. For a dynamic routing connection, connections initiated from the routing destination node can be used instead.

The AUTHENTICATED Routing User

If the connection URL specifies the SSL protocol, you can use the reserved word AUTHENTICATED as the username of a routing node. In this case, the connection will be made without testing for a password, and the SSL certificate identity will be used as the routing username.

If you choose a protocol other than SSL, you must specify a password for the routing username.

Logging

This section describes the two kinds of logging provided by SonicMQ:

- Server logs, which are optional text files that contain output that would otherwise be sent to the console
- Server recovery logs, which are binary files that allow a server to recover its state in the event of a system crash

Server Status/Error Log

A server log is a text file that is a redirection of server output that would normally be sent to the server console. This output includes, for example, startup and shutdown messages or Java stack traces of unexpected conditions that can occur on a system, such as a shortage of memory.

If you wish this log to be created and maintained, you must define the `BROKER_LOG` variable in the `broker.ini` file, specifying the pathname for this file.

A server log can be read by using Admin or Explorer or with any text editor. Admin and Explorer also provide you with the ability to clear the server log.

Server Recovery Logs

All server events are tracked in binary server recovery logs stored in the directory specified by the LOG_PATH property defined in the broker.ini file.

Server recovery logs enable a server to recover after a crash by using the server database and the contents of the server recovery log. If left to grow, server logs can become very large, depending on message traffic. Therefore, they are automatically limited to MAX_LOG_FILE_SIZE bytes. When the original server recovery log reaches this maximum size, new events are written to a second server recovery log. When the second server recovery log reaches this maximum size, new events are written to the first server recovery log, assuring that at least the last MAX_LOG_FILE_SIZE bytes of recovery log will always be available. By default, MAX_LOG_FILE_SIZE is 1,000,000, but you can change this value by setting it in the broker.ini file.

To enhance your ability to recover from a crash, you should periodically back up the server database and the server recovery logs. You must shut down the server before performing this backup.

JMS Administered Objects

As defined in the *Java Messaging Service, Version 1.0.2*, "**JMS administered objects** are objects containing JMS configuration information that are created by a JMS administrator and later used by JMS clients."

The JMS specification defines two alternative messaging domains:

- Publish and Subscribe (Pub/Sub), where one client produces a message and many clients can consume it
- Point-to-Point (PTP), where one client produces a message and one client consumes it

JMS defines two administered object types, **Destination** and **ConnectionFactory**. What they represent depends on the messaging domain:

- In the Publish and Subscribe domain:
  - Destination is a Topic
  - ConnectionFactory is a TopicConnectionFactory

- In the Point-to-Point domain:
  - Destination is a Queue
  - ConnectionFactory is a QueueConnectionFactory

The JMS specification does not spell out the details of Destinations or ConnectionFactories, so JMS administered objects from different vendors can be expected to have different structures.

Stored objects let the writer of a client application use symbolic names to refer to destinations and factories. The lookup name lets the client retrieve the data necessary to make connections to a server and publish and subscribe messages to queues and topics. You can use Admin or Explorer to set up the association between a lookup name and the data for the destination or factory. The factory data is the data needed to make a connection to the server. The destination data is the path to the topic or queue that is going to be used.

In the SonicMQ implementation, a Destination provides the following administrative information:

- Lookup name
- Destination type (topic or queue)
- Destination name (topic name in the Publish and Subscribe domain, queue name in the Point-to-Point domain)

In the SonicMQ implementation, a ConnectionFactory provides the following administrative information:

- Lookup name
- Factory type (TopicConnectionFactory or QueueConnectionFactory)
- URL list
- Default user
- Default password
- Connect ID
- Client ID
- Load balance flag
- List access flag (sequential or random)

Admin and Explorer each let you create and maintain SonicMQ-specific Destination and ConnectionFactory-administered objects. The object store can exist in either a file system or a Java Naming and Directory Interface (JNDI) name space.

Certificate Management

The Certificate Management Tool provides a set of commonly needed administration tools for managing the security certificates used in SSL-based network communications. It is part of the SonicMQ Explorer administration tool.

The Certificate Management Tool addresses several areas of administration for security certificates. For more information on security certificates, see the *SonicMQ Deployment Guide*.

You can perform the following tasks with the Certificate Management Tool:

- Generate RSA private/public key pairs.
- Generate a Certificate Signing Request (**CSR**), which you can then send to a Certificate Authority (**CA**).
- Load and store the Certificate Chain returned from the CA.
- Combine the private key with the Certificate Chain into PKCS12 format

A proprietary Certificate Store is used to store the certificate information for the Explorer. The Certificate Store contains of the following data:

- **Alias** — A unique ID that you assigned, which is used as an index for looking up the data within the Certificate Store.
- **SR data** — The key size and Relative Distinguished Name (RDN) data used to create the CSR.
- **Private Key** — The PKCS8-formatted RSA private key generated.
- **Certificates** — Either the PKCS7-formatted Certificate Chain returned from the CA or a single X509-formatted Certificate.

Chapter 3 Graphical Administration Tool

The **SonicMQ Explorer** provides a graphical user interface for administering the SonicMQ messaging system.

This chapter starts with a brief section describing how to start and stop the SonicMQ Explorer. The rest of the chapter describes how to use the tool. That section has three parts:

- "Message Servers" on page 109 describes the part of SonicMQ Explorer devoted to managing the servers.
- "JMS Administered Object Stores" on page 169 describes the part of SonicMQ Explorer devoted to managing JMS administered objects.
- "Certificate Management Tool" on page 177 describes the part of SonicMQ Explorer devoted to managing SSL certificates.

Starting and Stopping SonicMQ Explorer

You start Explorer in different ways depending on whether you have a Windows, UNIX, or Linux installation.

➤ **To start SonicMQ Explorer:**

- ☐ On Windows, choose **Start > Progress SonicMQ > Explorer**.
- ☐ On UNIX or Linux, run the `explorer.sh` script.

➤ **To stop SonicMQ Explorer:**

- ☐ Choose menu options **Explorer > Exit** or click the **Close** button for the window.

Customizing Explorer Behavior

You can modify the display behavior of SonicMQ Explorer by specifying:

- The maximum number of events displayed on the Explorer events node (See "Events" on page 148).
- The maximum number of messages held in the **Subscribed Messages** table for subscribers (See "Creating a Test Publish and Subscribe Session" on page 154) or receivers (See "Creating a Test Point-to-Point Session" on page 162).
- The number of points used to plot metrics (See "Metrics" on page 143).

You do this by setting the properties `admin.maxEvents`, `admin.maxMsgs`, and `admin.plotPoints` in the script that brings up Explorer (`explorer.bat` for Windows or `explorer.sh` for UNIX or Linux).

admin.maxEvents

`admin.maxEvents={`**`500`**`|`*`number_of_events`*`}`

Sets the maximum number of events displayed for a single server by Explorer on a first-in, first-out basis. The higher the value, the more memory is used.

admin.maxMsgs

`admin.maxMsgs={`**`50`**`|`*`number_of_messages`*`}`

Sets the maximum number of messages held in the Subscribed and Received Messages tables for Topic Subscribers and Queue Receivers (respectively). The higher the value, the more memory is used.

admin.plotPoints

`admin.plotPoints={`**`500`**`|`*`number_of_points`*`}`

Sets the number of plot points used by each plotted metric in Explorer. The default value provides sufficient plot points on the graph for most screen resolutions and refresh intervals. For very large screen resolutions you might wish to override the default. The higher the value, the more memory is used.

admin.savePrefs

admin.savePrefs={true|**false**}

If set to true, causes the panel configuration changes described below to be written to explorer.ini as they happen, so after you exit and re-enter Explorer, these changes are reflected in the GUI:

- **Metrics Node** — On the Metrics Node, the settings saved are those on the Summary tab, whether a metric has a single view graph and whether it is included in the multiple view graph tab.
- **Events Node** — On the Events Node, the settings saved are those that indicate whether a particular event is being tracked.

admin.reqTimeout

admin.reqTimeout={**30**|*seconds*}

Sets the number of seconds that Explorer waits for a response to a request before timing out. If a timeout occurs, an error dialog box is displayed.

Setting Explorer Properties

You can set any of the Explorer startup properties by prefacing the property with -D and placing it before the name of the class progress.message.tools.Explorer in the explorer script Java command.

For example, suppose you wish to change admin.maxEvents to 1000.

On Windows, change the line in explorer.bat from

```
"%SONICMQ_JREW%" -cp "%JRE_CLASSPATH%;%SONICMQ_CLASSPATH%;" \
progress.message.tools.Explorer
```

to

```
"%SONICMQ_JREW%" -cp "%JRE_CLASSPATH%;%SONICMQ_CLASSPATH%;" \
-Dadmin.maxEvents=1000 progress.message.tools.Explorer
```

On UNIX or Linux, change the line in explorer.sh from

```
$SONICMQ_JRE -classpath "$JRE_CLASSPATH:$SONICMQ_CLASSPATH" \
progress.message.tools.Explorer
```

to

```
$SONICMQ_JRE -classpath "$JRE_CLASSPATH:$SONICMQ_CLASSPATH" \
-Dadmin.maxEvents=1000 progress.message.tools.Explorer
```

Using SonicMQ Explorer

When you start SonicMQ Explorer for the first time, you see a window similar to the one shown in Figure 8.

The SonicMQ Explorer window has two panels. The left (tree) panel displays **Certificate Stores**, **JMS Administered Object Stores,** and **Message Brokers** folders, and the contents of each, in a tree structure. The right panel displays information and controls that relate to the currently selected node in the tree panel.

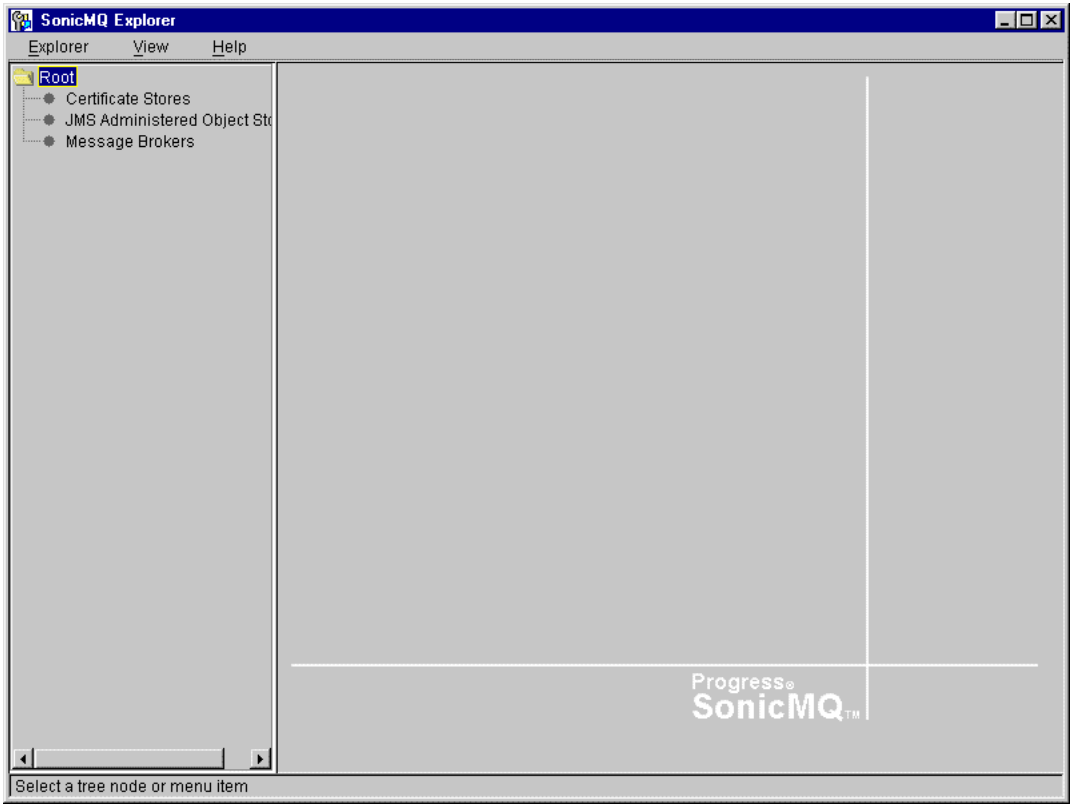


Figure 8. Initial SonicMQ Explorer Window

Server data is represented as nodes under **Message Brokers** and stored in the database for each server.

Certificate Stores are represented as nodes under **Certificate Stores**. These are files that you name and that store security certificates, keys, and related data. JMS administered objects are represented as nodes under **JMS Administered Object Stores**. They are stored in object stores, either in a file system directory that you specify or under a JNDI Naming Service that you specify.

Message Servers

Select **Message Brokers** in the tree panel to display information and options for servers.

A **New Connections** panel appears on the right, as shown in Figure 9.

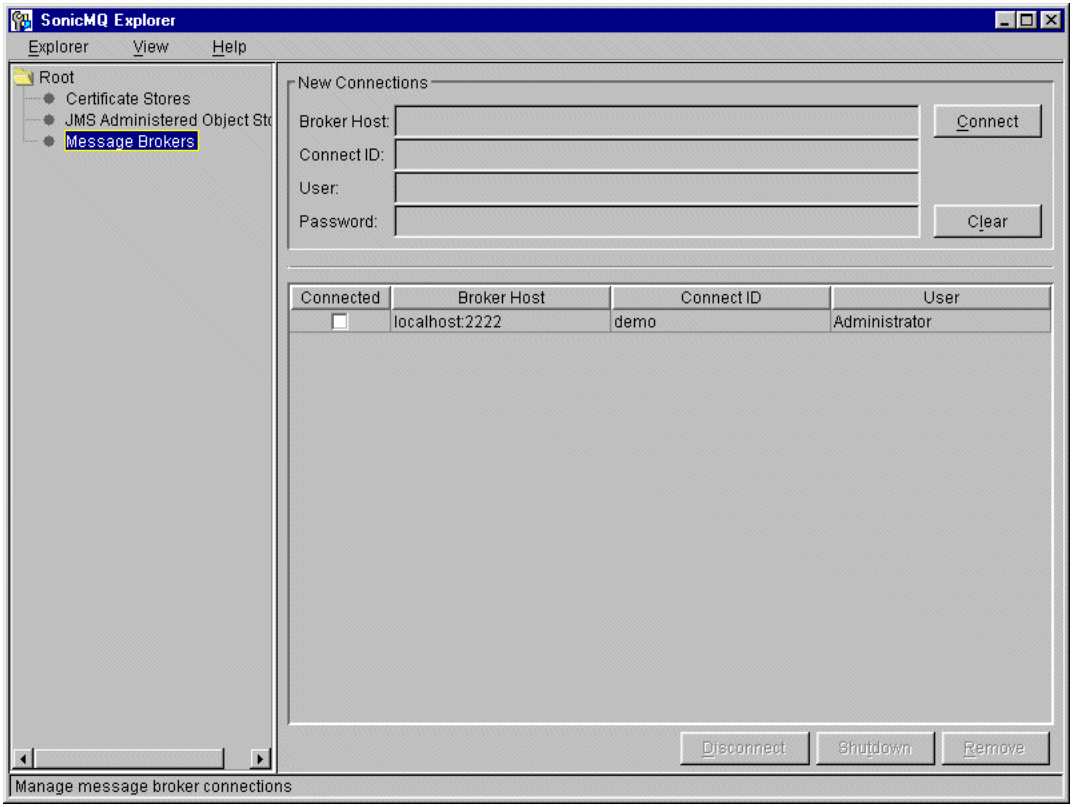


Figure 9. New Connections

Once you have connected to a server, a list of the connected servers appears in the tree in the left panel and in the table as shown in Figure 10 on page 112.

Connecting to a Server

To connect to a server, you specify the information required in the **New Connections** dialog box and then click **Connect**. The following information is required:

- **Broker Host** — A URL to the message server in the form {**tcp://**|http://|ssl://}*host_name*{**:2506**|:*port*}, where the protocol (tcp, http, or ssl) is only necessary if you are specifying a protocol other than tcp and *port* is only necessary if you are specifying a nondefault port (one other than 2506)
- **Connect ID** — Any convenient text string, for example ADMIN, used to name a JMS client connection to the server in a queue or topic session
- **User** — A username required for a security-enabled server
- **Password** — A password associated with **User,** required for a security-enabled server

The information you enter (except the password) appears in the connection table in the bottom-right panel.

Note If you installed SonicMQ following the automated installation procedure, you set **Broker Host** to localhost (or your actual hostname) and set a **Connect ID**. The server is not security-enabled, so you need not enter **User** or **Password**. The server is also not interserver-enabled.

While connecting to servers, Explorer maintains a list of known hosts, Application IDs, and users in the explorer.ini file of the working directory. This data is known as the **prior connections table**. The prior connections table is read when Explorer starts to initialize the connection table.

If data already exists in the prior connections table when you start Explorer, you can select an entry in the table, and the data is copied to the **New Connections** panel. Enter the password (if the server is security-enabled) and click **Connect** to establish a connection.

You can clear all fields in the **New Connections** dialog box by clicking **Clear**.

When first started, a security-enabled server has a default user, `Administrator`, with a default password, also `Administrator`. You should change the password as soon as possible for security reasons. (See "To change a user's password:" on page 120.)

Once you have connected to a server, you can perform a large number of functions, which are discussed in the remainder of this chapter. In particular from the **Message Brokers** panels, you can:

- Disconnect a server from Explorer by selecting a connected server in the connections table and clicking **Disconnect**. You can subsequently reconnect the server.
- If you are a member of the Administrators group or security is not enabled, you can shut down a server by selecting a connected server in the connections table and clicking **Shutdown**.

  If you shut down a server or if it can be detected that a server has terminated (shut down by another administrator or otherwise), the server is removed from the list of servers shown in the tree.
- Remove a disconnected server from the connections table by selecting the server in the table and clicking **Remove**. The entry is removed from the Explorer initialization file `explorer.ini`.

Server Tree Nodes

As shown in Figure 10, child nodes appear for each server in the tree.

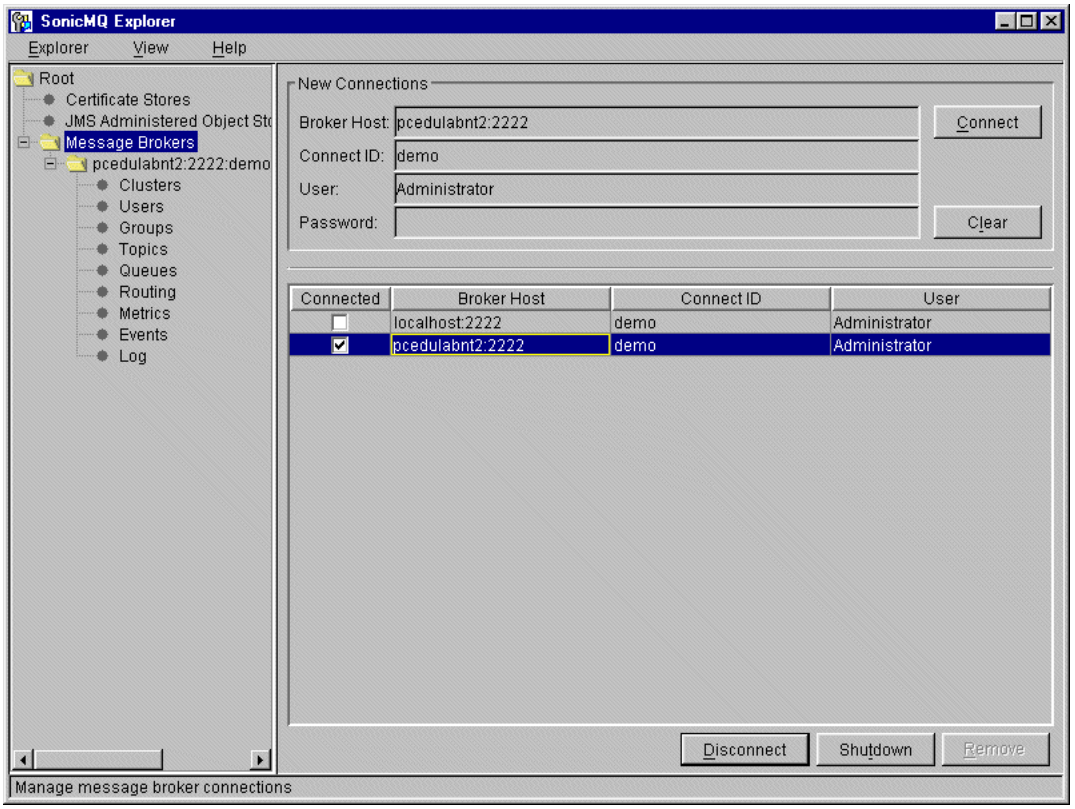


Figure 10. Message Servers

The following child nodes might appear for each server in the tree:

- **Clusters**
- **Users**
- **Groups**
- **Topics**
- **Queues**
- **Routing**
- **Metrics**

- **Events**
- **Log**

You can manage the objects at each node by selecting the node and using the tools provided in the right panel. Each node is described in the following sections.

Depending on the way you initialized your server and the privileges of the connected user, various child nodes might or might not appear. Nodes do not appear unless the connected user has Administrator privileges (or security is not enabled). If this condition is met, a **Metrics** node, a **Routing** node, and an **Events** node appear. In addition:

- **Clusters** appear only if the server is an interserver configuration server; that is, the following two conditions are met:
  - `ENABLE_INTERBROKER=TRUE` appears in the `broker.ini` file for the server.
  - `IB_CONFIG_SERVER` is not set, which is the case for the configuration server. (`IB_CONFIG_SERVER` must, however, be set on all other servers that are candidates for cluster membership.)
- **Groups** and **Users** appear only if the following conditions are met:
  - The server is security-enabled; that is, `ENABLE_SECURITY=TRUE` appears in the `broker.ini` file for the server.
  - The server is a configuration server or a stand-alone server.
- **Topics** and **Queues** always appear, providing the following functions:
  - **Topics** node — A tab to manage durable subscriptions
  - **Queues** node — Tabs to manage queues and messages

  If the server is security-enabled and the server is either a configuration server or a stand-alone server, these nodes also provide tabs to maintain topic and queue security.

Note To manage durable subscriptions and queues, you must always connect to the server that hosts the specified subscriptions and queues. This is true even if the server is part of a cluster. Thus, you cannot use the interserver configuration server to manage the durable subscriptions and queues hosted by other servers that are members of the cluster.

- **Log** appears only if the server is log-enabled; that is, `BROKER_LOG=`*`logpath`* appears in the server's `broker.ini` file, where *`logpath`* is the relative or absolute pathname of the server log file.

Note **Topics** and **Queues** functions are restricted unless the server is a security-enabled configuration server or security-enabled stand-alone server, and the connected user has Administrator privileges. The right panel display for **Topics** or **Queues** differs, depending on whether their functions are restricted. (See "Topics" on page 127 and "Queues" on page 131.)

By selecting a node object, editable or status information on the object is displayed in the right panel. Data for the node is read once on connect, then each time you select the **View > Refresh** menu item (or the **Refresh** button, where available). You can manipulate the tree using the expand and collapse icons.

Important Do not use these characters in user, group, server, or cluster names: period (`.`), asterisk (`*`), pound sign (`#`), backslash(\), or dollar sign (`$`).

Clusters

Clusters can only be created in the SonicMQ Professional Developer and E-Business Editions. A cluster is a group of servers that are connected to one another. See "Clustered Servers" on page 83 for an overview of clusters.

When you select the **Clusters** node for a server in the tree panel, the right panel displays two tabs, **Clusters** and **Brokers**, as shown in Figure 11.

Clusters Tab

The **Clusters** tab shows a list of clusters and a list of member servers for a cluster you select.

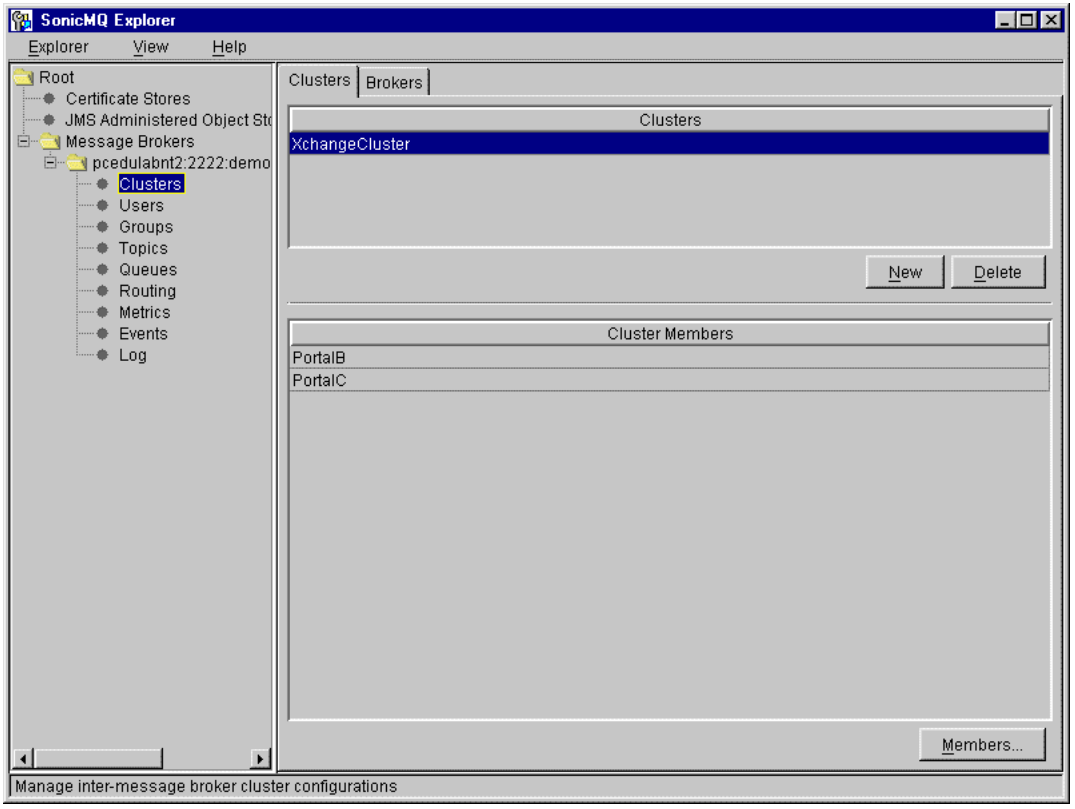


Figure 11. Clusters

Use this window to create, change, or delete a cluster.

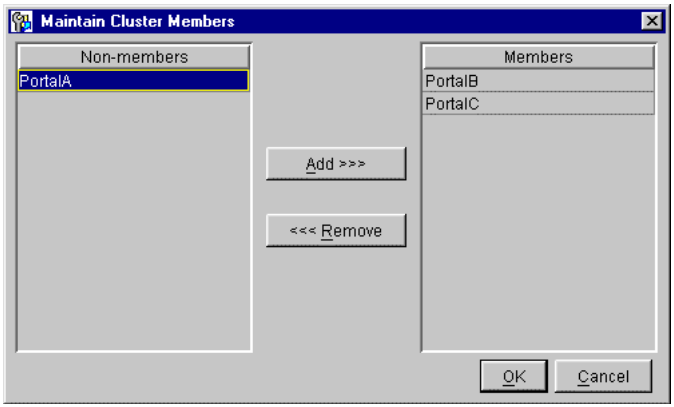
➤ **To create a new cluster:**

1. Click **New**. A blank field appears at the bottom of the **Clusters** list.
2. Type the name of the cluster in the blank field and press **Enter**.

Important The name of a cluster can contain a maximum of 64 Unicode characters and cannot contain the characters period (.), asterisk (*), pound sign (#), backslash(\), or dollar sign ($).

➤ To add or remove servers in a cluster:

1. Select the cluster and click **Members**. The **Maintain Cluster Members** dialog box appears:



   If you have just created the cluster, all candidate servers known by the configuration server appear in the **Non-members** list on the left.

2. Add servers:

Important

Before adding a server to a security-enabled interserver cluster, you must perform several preliminary steps. See "Adding Servers to Security-enabled Clusters" on page 81 for details.

   2.1 Select servers in the **Non-members** list.

   2.2 Click **Add** to move them to the **Members** list on the right.

3. Remove servers:
   - 3.1 Select servers in the **Members** list.
   - 3.2 Click **Remove** to move them to the **Non-members** list on the left.
4. Click **OK** when you are done (or **Cancel** to exit without taking action).

A cluster can only be deleted if its member servers have previously been removed from the cluster.

➤ **To delete an existing empty cluster:**

1. Select the cluster in the **Clusters** list.
2. Click **Delete**.

Brokers Tab

When you select the **Brokers** tab, you see a list of servers that are available for use in clusters, as shown in Figure 12. You can remove servers from the list by selecting the server and clicking **Delete**. You cannot delete the configuration server itself and you cannot delete a server if it is running or already part of a cluster.

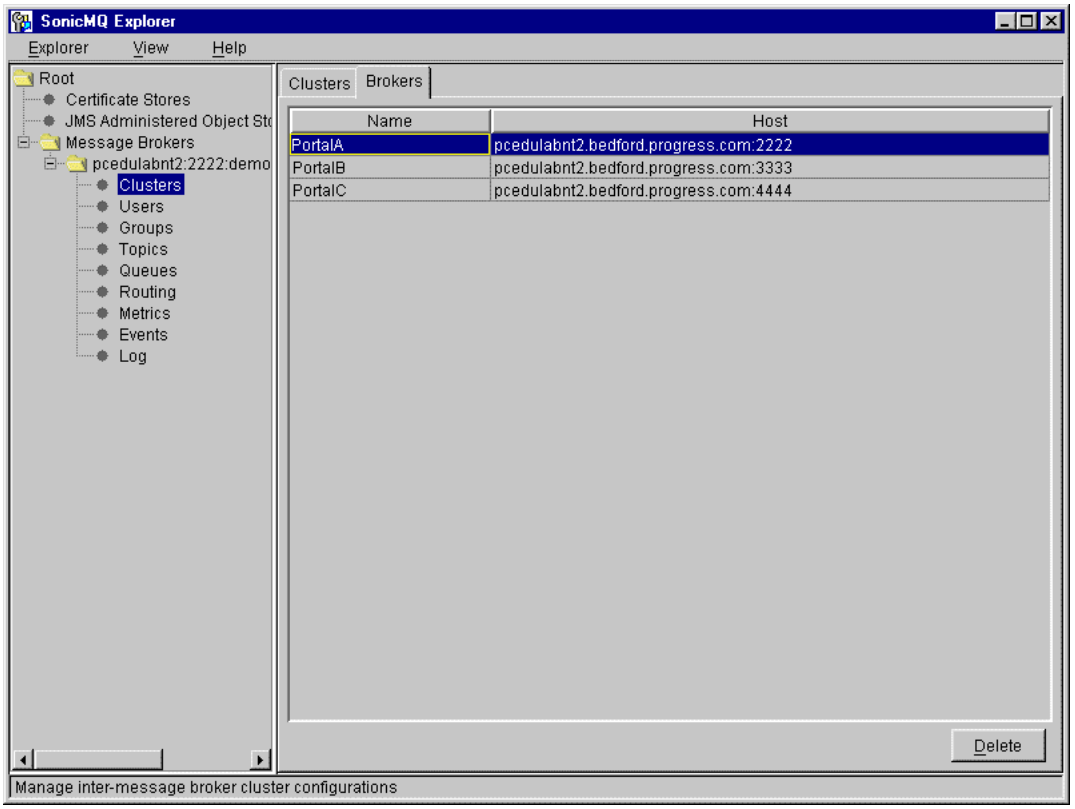


Figure 12. Clusters: Servers

Users

When you select the **Users** node for a server in the tree panel, the right panel displays three tabs, **Users**, **Routing Users**, and **Group Memberships**, as shown in Figure 13.

Users Tab

The **Users** tab, displayed by default, allows you to see the users listed by name.

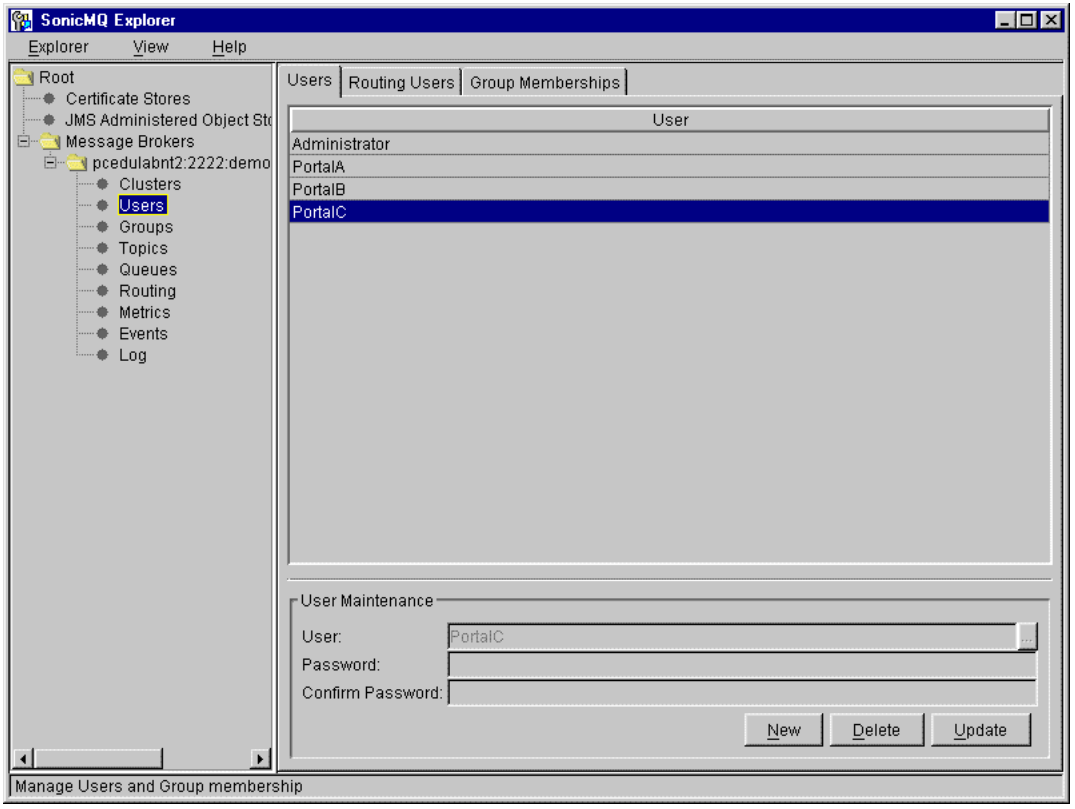


Figure 13. Users

Note If the server being maintained is a security-enabled configuration server, all servers known by the configuration server appear as users in the **Users** list.

Important Do not create a user with the same name as a server. SonicMQ creates users with the same names as servers when setting up interserver (cluster) configurations. These are special users that are added to the Administrators group and should not be used for normal client connections using Explorer, the Admin tool, or JMS application code.

Use this window to add or delete a user, or to change a user's password.

➤ **To add a user:**

1. Click **New**.
2. Type the username in the **User** field and press **Enter**. The username can have a maximum of 64 Unicode characters. Do not use the characters period (`.`), asterisk (`*`), pound sign (`#`), backslash(`\`), or dollar sign (`$`) in a username. Do not use the reserved name `AUTHENTICATED`.
3. Optionally, type in the **Password** field and press **Enter** and retype the same password in the **Confirm Password** field and press **Enter** again.
4. Click **Update**.

➤ **To change a user's password:**

1. Select the user's name in the **User** list.
2. Fill in the **Password** field with the new password and press **Enter**. An asterisk (`*`) appears in place of each character typed.
3. Fill in the **Confirm Password** field with the new password and press **Enter**. An asterisk (`*`) appears in place of each character typed.
4. Click **Update**.

➤ **To delete a user:**

1. Select the user's name in the **User** list.
2. Click **Delete**.

Routing Users Tab

When you select the **Routing Users** tab, the right panel of the window shows a **Routing Users** tab. (See Figure 14.) The **Routing Users** tab displays a list of users and their associated routing nodes managed by the server. A routing user's user name is used for authenticating routing connections between servers. The routing node is the node where the inbound connection originates.

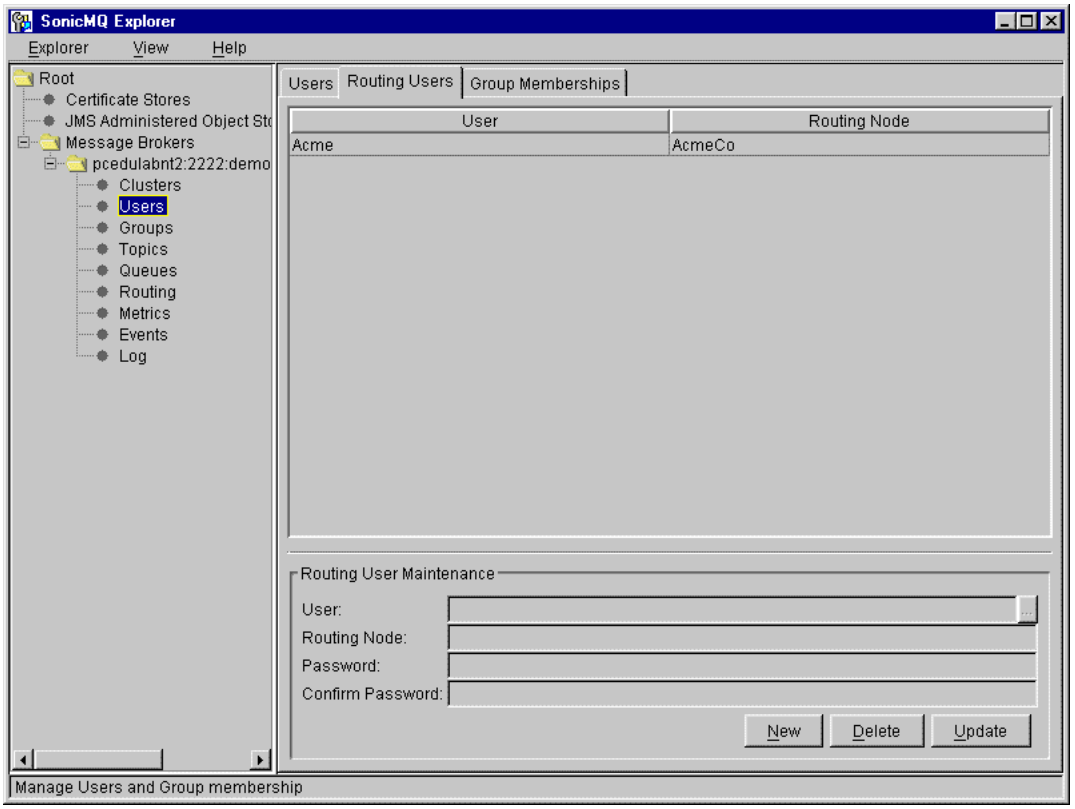


Figure 14. Users: Routing Users

You can add a user to the list of routing users managed by the security-enabled server.

➤ **To add or update a routing user:**

1. Click **New**.
2. Select a user.
3. Enter the routing node name from which the inbound connection originates.
4. Optionally, type in the **Password** field and press **Enter** and retype the same password in the **Confirm Password** field and press **Enter** again.

   The routing user is assigned the specified password if you supply it, otherwise the user is assigned a blank password. The password is not checked over an SSL connection when the user name is the certificate identity.
5. Click **Update**.

➤ **To delete a routing user:**

1. Select the user's name in the **User** list.
2. Click **Delete**.

   SonicMQ deletes the existing routing user from the list of users managed by the server.

Note **Users** and **Routing Users** names must be unique. If you try to create a new user with an existing routing user name, or you try create a new routing user with an existing user name, Explorer displays an error message.

Group Memberships Tab

When you select the **Group Memberships** tab, the right panel of the window shows a **Group Memberships** tab. (See Figure 15.) This tab displays a list of all users and their associated group memberships. (For instructions on how to create and manage groups, see "Groups" on page 125.)

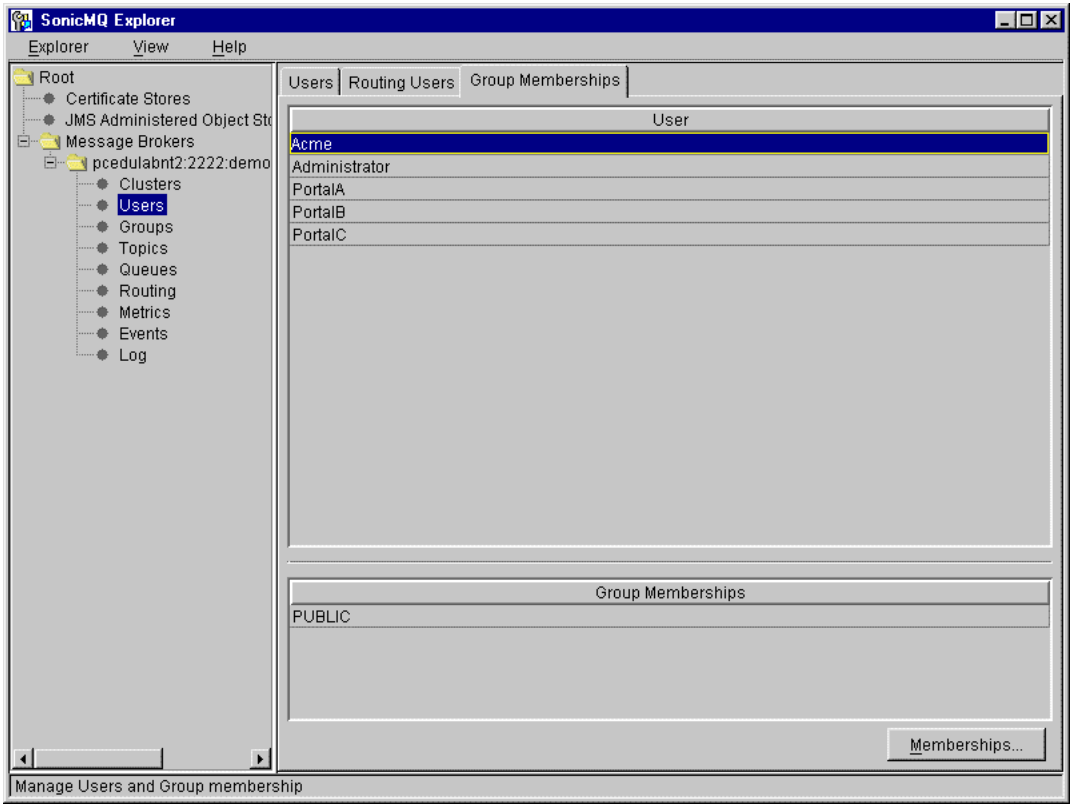


Figure 15. Users: Group Memberships

You can use the **Group Memberships** tab for several functions.

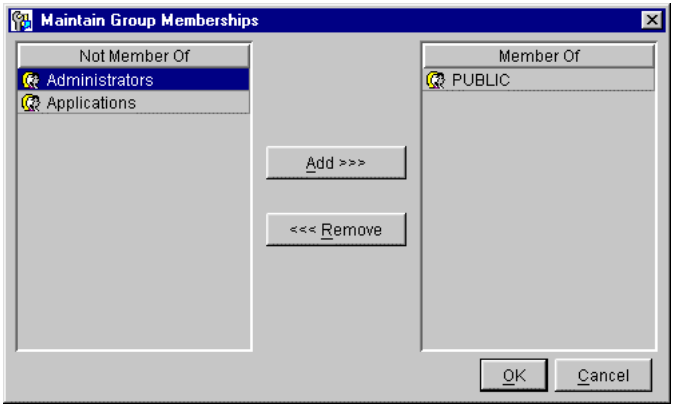
➤ **To display the groups a user belongs to:**

☐ Select the user in the **User** list.

The groups the user belongs to are displayed in the **Group Memberships** list.

➤ To add a user to a group or remove a user from a group:

1. Select the user.
2. Click **Memberships**. The **Maintain Group Memberships** dialog box appears:



3. To add the user to groups:
   - 3.1 Select the groups in the **Not Member Of** list.
   - 3.2 Click **Add**.
4. To remove the user from groups:
   - 4.1 Select the groups in the **Member Of** list.
   - 4.2 Click **Remove**.
5. Click **OK** when you are done (or **Cancel** to exit without taking action).

Groups

When you select the **Groups** node for a server in the tree panel, the right panel shows a list of groups, together with the members of a given group. (See Figure 16.) You can add or delete members from a group, and you can add or delete groups.

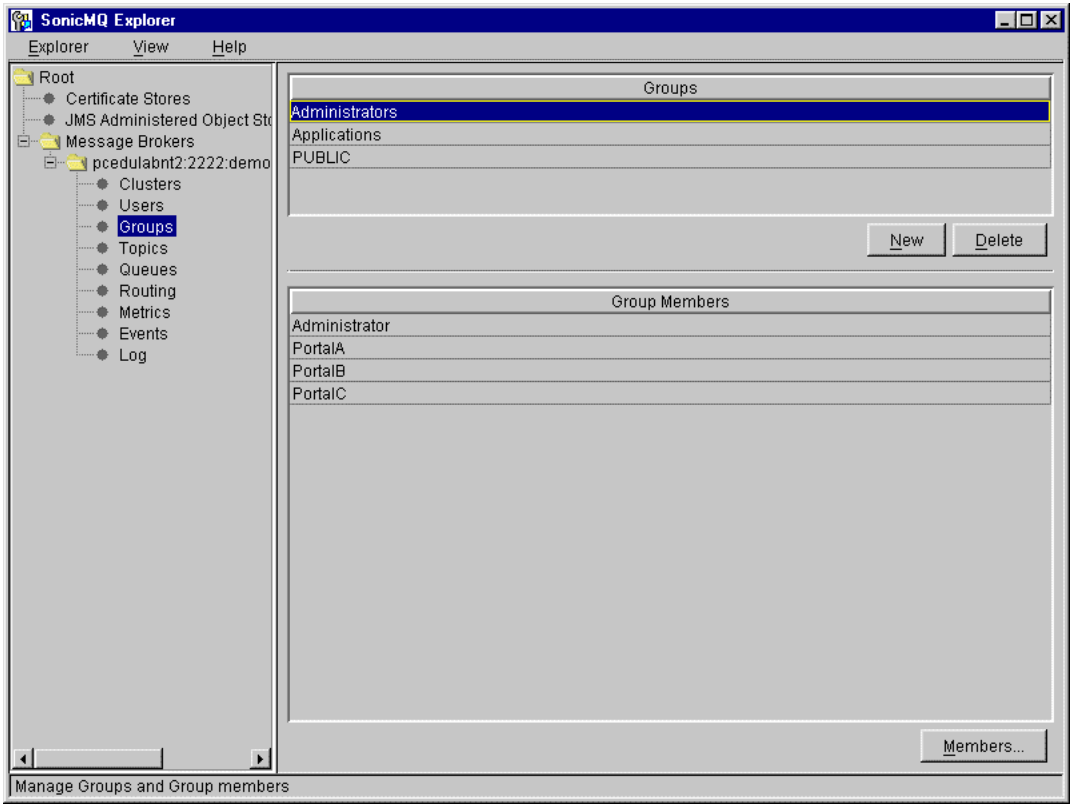


Figure 16. Groups

You can use this window for several functions.

➤ **To display the members in a group:**

☐ Select the group in the **Groups** list.

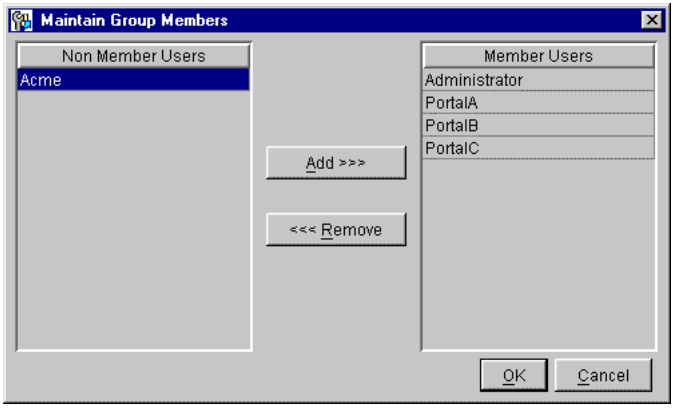
➤ To add a new group:

1. Click **New**. A blank field appears in the **Groups** list.
2. Type the name of the group in the blank field and press **Enter**. The name can have a maximum of 64 Unicode characters. Do not use the characters period (**.**), asterisk (*****), pound sign (**#**), backslash(\), or dollar sign (**$**) in a group name.

   The group is created when you move focus from the new row.

➤ To add or remove members from a group:

1. Select the group and click **Members**. The **Maintain Group Members** dialog box appears:



   If you have just created the group, all users appear in the **Non Member Users** list on the left.

2. Add users:
   - **2.1** Select the users in the **Non Member Users** list.
   - **2.2** Click **Add** to move them to the **Member Users** list on the right.
3. Remove users:
   - **3.1** Select the users in the **Member Users** list.
4. Click **Remove** to move them to the **Non Member Users** list on the left.
5. Click **OK** when you are done (or **Cancel** to exit without taking action).

➤ **To delete an existing group:**

1. Select the existing group in the **Groups** list.
2. Click **Delete**.

Topics

When you select the **Topics** node for a server in the tree panel, the right panel displays a **Durable Subscriptions** tab and, possibly, a **Topic Security** tab.

Topic Security Tab

The **Topic Security** tab is visible if the server is a security-enabled stand-alone server or a security-enabled configuration server.

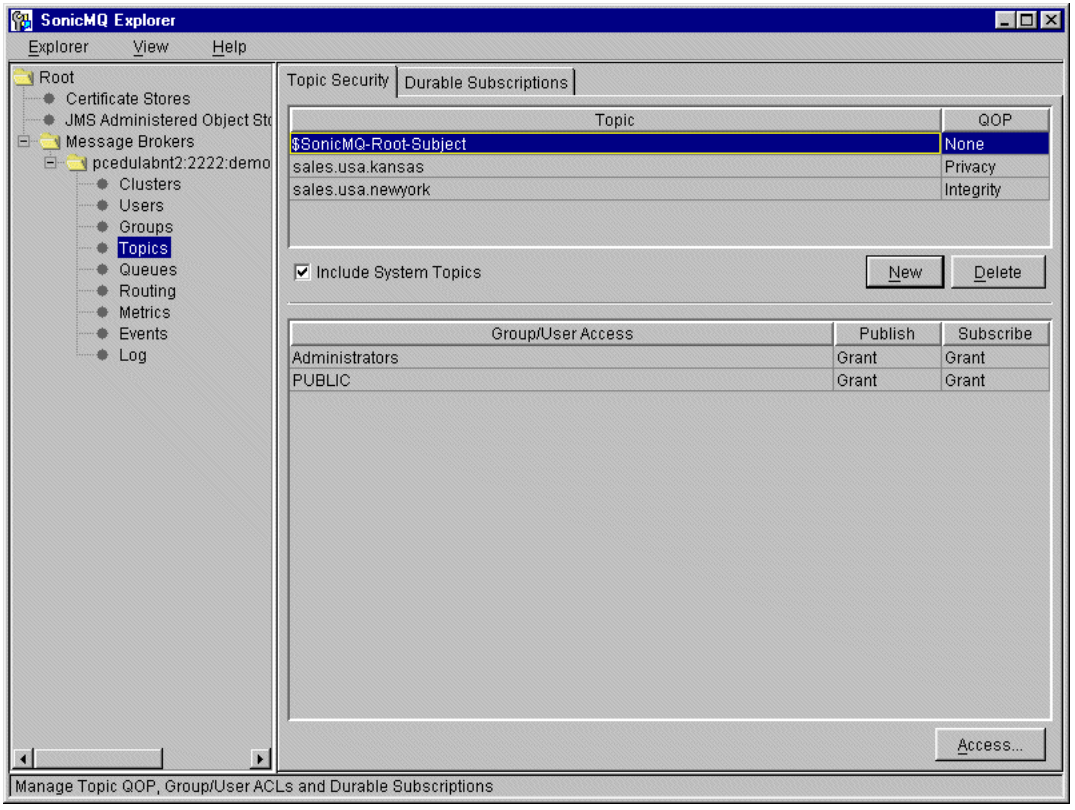


Figure 17. Topics (Security-enabled)

The **Topic Security** tab allows you to manage topics and topic security. (See Figure 17.) You can add or delete topics. For each topic, you can see and specify the following security properties:

- Quality of Protection (QoP)
- Access Control List (ACL)

To maintain an ACL, you can grant or deny users and groups access to a topic for either or both publishing and subscribing. You can also specify that users inherit their access permissions from the groups to which they belong. (See "Groups and Security" on page 98 for details.)

Use the **Topic Security** tab to add or delete a topic, or to change QoP settings.

➤ To show system topics:

By default, SonicMQ does not show system (internal) topics.

☐ Select the **Include System Topics** check box to display the system topics.

➤ To add a topic:

1. Click **New**. A new row appears in the **Topic** table.
2. Type the name of the topic in the **Topic** field of the new row. The name can contain up to 256 Unicode characters. It cannot contain the following characters:
   - Backslash (\)
   - Double colon (`::`)
   - Dollar sign (`$`)
   - Wildcard characters (`*` or `#`), except as wildcards

   A topic name cannot begin with the string "SonicMQ.".

   You can use dot-delimited hierarchy naming conventions when naming topics. See "Inheritance of Security Policies" on page 95 and "Security Policies and Name Spaces" on page 97 for more information.
3. Select the desired QoP option from the drop-down list to the right of the topic name.
4. Press **Enter**.

➤ To change the QoP option for a topic:

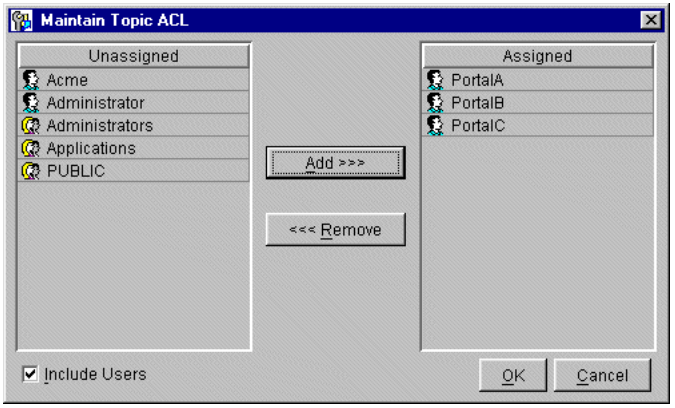
1. Select the **Topic** in the upper list.
2. Select the desired QoP option from the drop-down list to the right of the topic name. The QoP options available are **Privacy**, **Integrity**, and **None**. See "Quality of Protection" on page 87 for a description of these options.

➤ To delete a topic:

1. Select the topic to be deleted.
2. Click **Delete**.

➤ To create or modify an Access Control List:

1. Click **Access**. The **Maintain Topic ACL** dialog box appears:



2. To add groups to an ACL, select the groups from the **Unassigned** list and click **Add**.
3. To add users to an ACL, select **Include Users** to show users as well as groups in the **Unassigned** list, then select the users and click **Add**.

   You can differentiate between groups and users by the icon to the left of each name. The single-face icon indicates a user, and the double-face icon indicates a group.
4. To remove users and/or groups from an ACL, select the users and/or groups from the **Assigned** list and click **Remove**.
5. When you are done, click **OK**.

Durable Subscriptions Tab

When you select the **Durable Subscriptions** tab, the Durable Subscription tab displays, as shown in Figure 18.

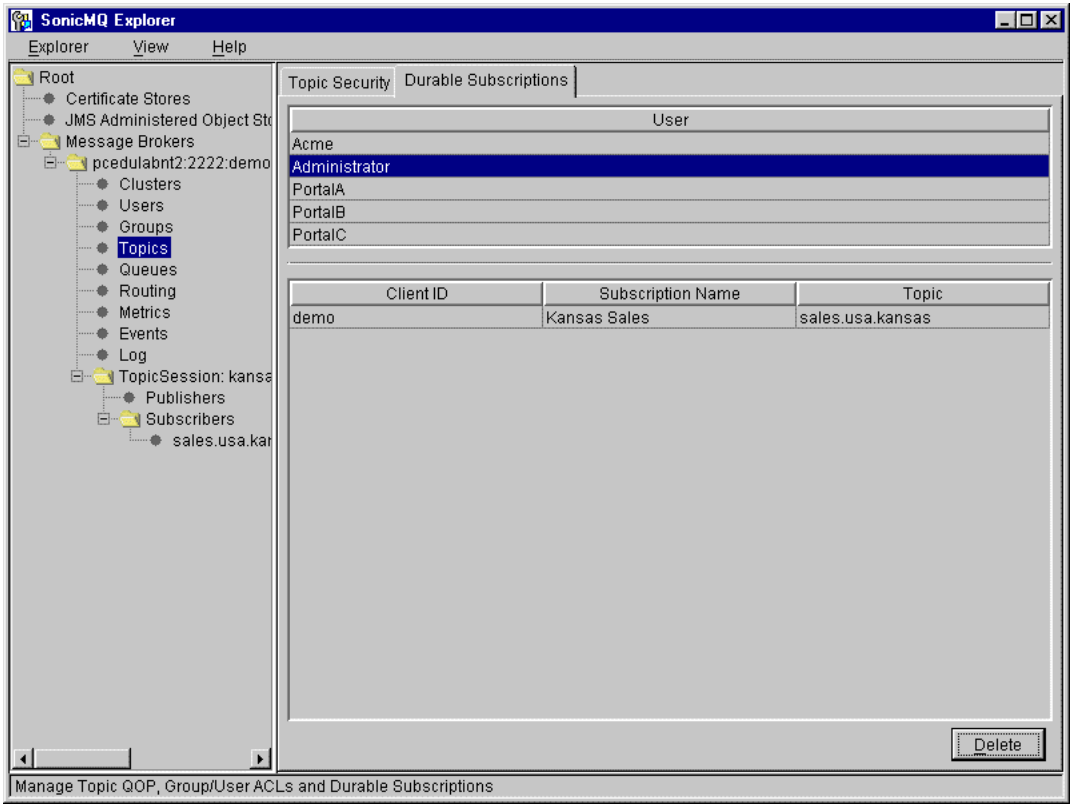


Figure 18. Topics: Durable Subscriptions

This tab enables you to view and delete durable subscriptions. If security is enabled, the list of users in the upper-right panel includes all users irrespective of whether they have durable subscriptions. If security is not enabled, only users that currently have durable subscriptions are listed.

Note To create a durable subscription using Explorer, you must create a Publish and Subscribe session and then create a durable subscriber from the **Subscribers** node of the session. See "Creating a Test Publish and Subscribe Session" on page 154 for details.

➤ **To view the durable subscriptions for a user:**

- ☐ Select a user in the upper table. The lower table shows the durable subscriptions that the user has created, by Client ID and Subscriber Name.

➤ **To delete durable subscriptions for a user:**

1. Select the user in the upper table.
2. Select one or more durable subscriptions in the lower table.
3. Click **Delete**.

Queues

When you select the **Queues** node for a server in the tree panel, the right panel displays two or three tabs. If the server is a security-enabled stand-alone server or a security-enabled configuration server, the tabs **Queues**, **Queue Security**, and **Messages** appear as shown in Figure 19.

If the server is not a security-enabled stand-alone server or a security-enabled configuration server, only the **Queues** and **Messages** tabs appear.

Queues Tab

The **Queues** tab allows you to manage physical queues and lists the queues currently hosted by the connected server.

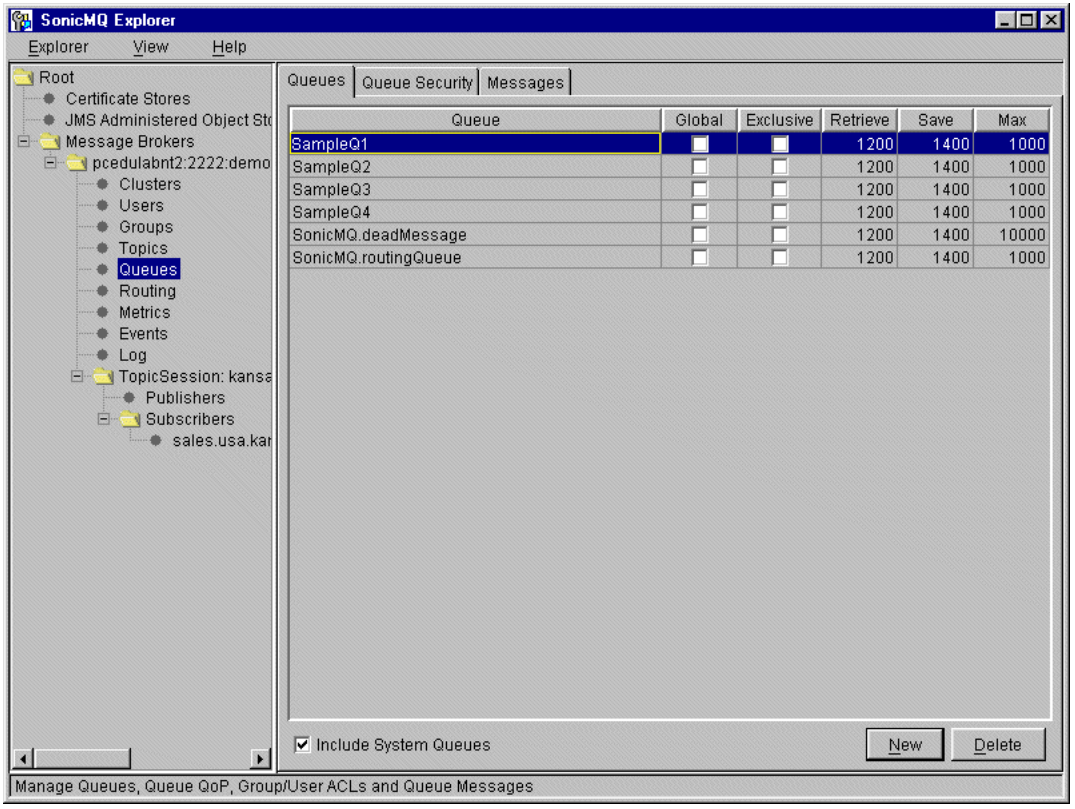


Figure 19. Queues (Security-enabled)

➤ To show system queues:

By default, SonicMQ does not show system (internal) queues.

- ☐ Select the **Include System Queues** check box to display the system queues, including the dead message queue.

➤ **To add a new queue:**

1. Click **New**. A new row appears in the **Queue** table.
2. Type the name of the queue in the **Queue** field of the new row. The name can contain up to 256 Unicode characters. It cannot contain the following characters:
   - Backslash (\)
   - Double colon (**::**)
   - Dollar sign (**$**)
   - Wildcard characters (***** or **#**)

   A queue name cannot begin with the string “SonicMQ.”.

   You can use dot-delimited hierarchy naming conventions when naming queues. See “Inheritance of Security Policies” on page 95 and “Security Policies and Name Spaces” on page 97 for more information.
3. Change the properties of the queue as you want.
4. Press **Enter**.

The **Queues** tab also lets you regulate the virtual memory consumed by a queue and the concurrent access to a queue. The **Retrieve, Save**, and **Max** options implement virtual memory for the queue. The **Exclusive** option, when set, specifies that only one user can read from the queue. If more than one user has permission to receive from a read-exclusive queue, the first user to access the queue maintains exclusive read privileges as long as the user remains connected.

There are restrictions on system queues. For example, you cannot make the routing queue exclusive, and you cannot make the routing queue or the dead message queue global.

Note To guarantee that enqueued messages are received in the same order they are put on a queue, you must make the queue read-exclusive.

➤ **To change queue attributes:**

1. Check **Global** to make the queue global. (A global queue allows other routing servers to write to it. A local queue does not allow other routing servers to write to it. By default, a queue is local.) Uncheck **Global** to make the queue local.
2. Check **Exclusive** to make the queue read-exclusive. Uncheck **Exclusive** to make it not read-exclusive.
3. Click **Save,** overtype the field, and press **Enter** to change the value.

   Save, short for Save Threshold, is the maximum total size of messages, in KB, that can reside in memory at one time. As additional messages are sent to the queue, they are saved in the database.
4. Click **Retrieve**, overtype the field, and press **Enter** to change the value.

   Retrieve, short for Retrieve Threshold, is a positive number less than or equal to Save Threshold. When the total size of messages, in KB, stored in memory is less than or equal to Retrieve Threshold, messages in the database are downloaded into memory, subject to the restriction that there can be no more than Save Threshold KB of messages in memory.
5. Click **Max**, overtype the field, and press **Enter** to change the value.

   Max is the maximum total size of enqueued messages, in KB. If a queue receives a message that causes its maximum capacity to be exceeded, publishing to the queue is delayed until the total queue size drops below the maximum.

➤ **To delete a queue:**

1. Select the queue to be deleted.
2. Click **Delete**.

Queue Security Tab

When you select the **Queue Security** tab, the top panel contains a list of logical queues and the bottom panel contains the Access Control List for the selected queue, as shown in Figure 20. A **logical queue** identifies an individual physical queue for which you want to specify nondefault security. You can use this tab to assign Quality of Protection and access control to individual queues or to all queues in a given name space.

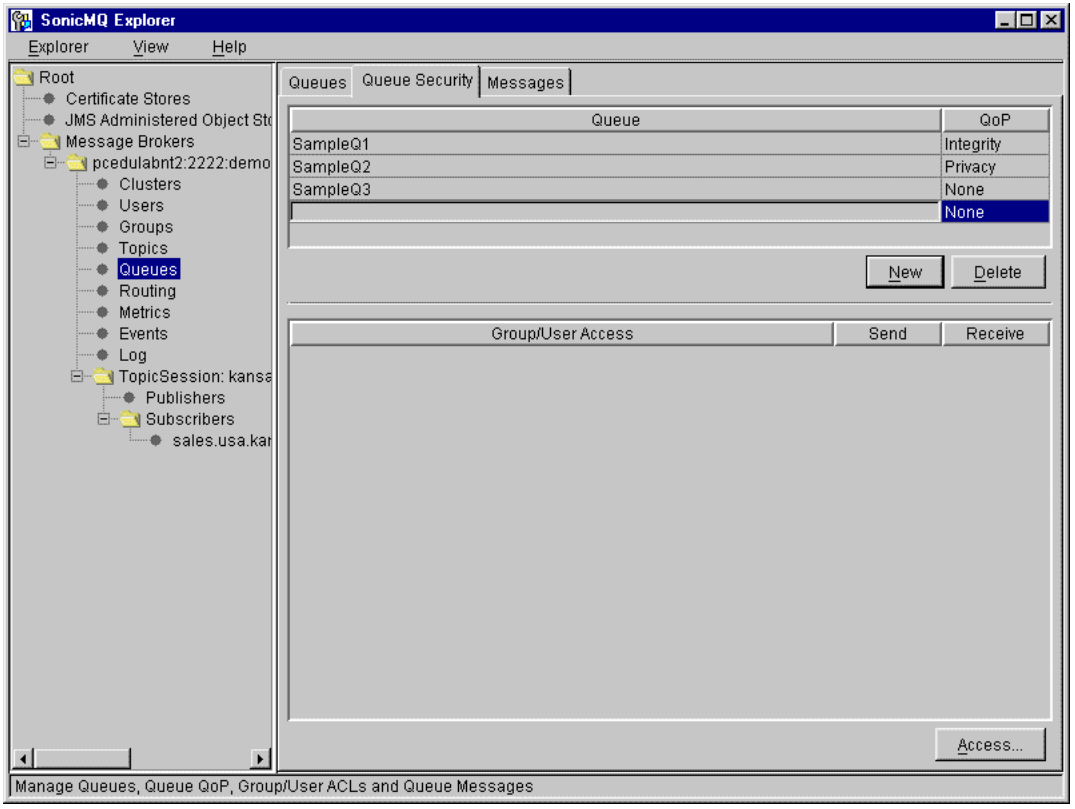


Figure 20. Queues: Queue Security

Note Physical queues created on the **Queues** tab do not automatically appear on the **Queue Security** tab. Instead, they inherit the default global security behavior. To specify nondefault security, you must define a logical queue or queue namespace on the **Queue Security** tab.

➤ To add the name of a new queue or name space:

1. Click **New**. A blank line appears in the **Queue** table in the upper-right panel.
2. Enter the queue name or the name space. A name space must include the asterisk character (*) or the pound character (#).
3. Select the desired Quality of Protection from the **QoP** drop-down list.
4. Press **Enter**.

➤ To change the QoP option for a queue:

1. Select the **Queue** in the upper list.
2. Select the desired QoP option from the drop-down list to the right of the queue name. The QoP options available are **Privacy**, **Integrity**, and **None**. See "Quality of Protection" on page 87 for a description of these options.

➤ To delete a logical queue or queue name space:

1. Select the logical queue or name space to be deleted.
2. Click **Delete**.

➤ To assign access to a queue or queue name space:

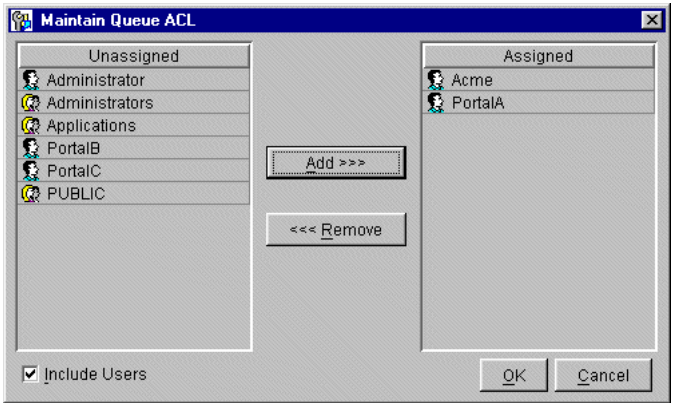
1. Select the queue or name space in the upper-right panel.
2. Click **Access**. A **Maintain Queue ACL** dialog box appears, identical to the one used with the **Queues** tab, shown on page 137. Principals appear in the **Unassigned** list. Select **Include Users** if you wish to see users (as well as groups) in the **Unassigned** list.
3. To move users or groups between the **Unassigned** list and the **Assigned** list, select the items and click the **Add** or **Remove** button, as required.
4. When done, click **OK** to confirm or click **Cancel**.

➤ To change access settings for users:

1. Select the queue or name space.
2. Select the **Send** or **Receive** column for the group (or user) that you wish to change. Use the drop-down list to change the value, which can be:
   - **Inherit**
   - **Grant**
   - **Deny**
3. Repeat Step 2 for each group (or user) as required.

➤ To create or edit the Access Control List for a queue:

1. Select the queue.
2. Click **Access**. The dialog box shown below appears with the principals (users and groups) in the **Unassigned** list:



3. Select **Include Users** if you wish to see users (as well as groups) in the **Unassigned** list.
4. To move users or groups from one list to the other, select the items and click the **Add** or **Remove** button, as required.
5. When done, click **OK** to confirm or click **Cancel**.

Messages Tab

When you select the **Messages** tab, the right panel displays a table of queues, as shown in Figure 21.

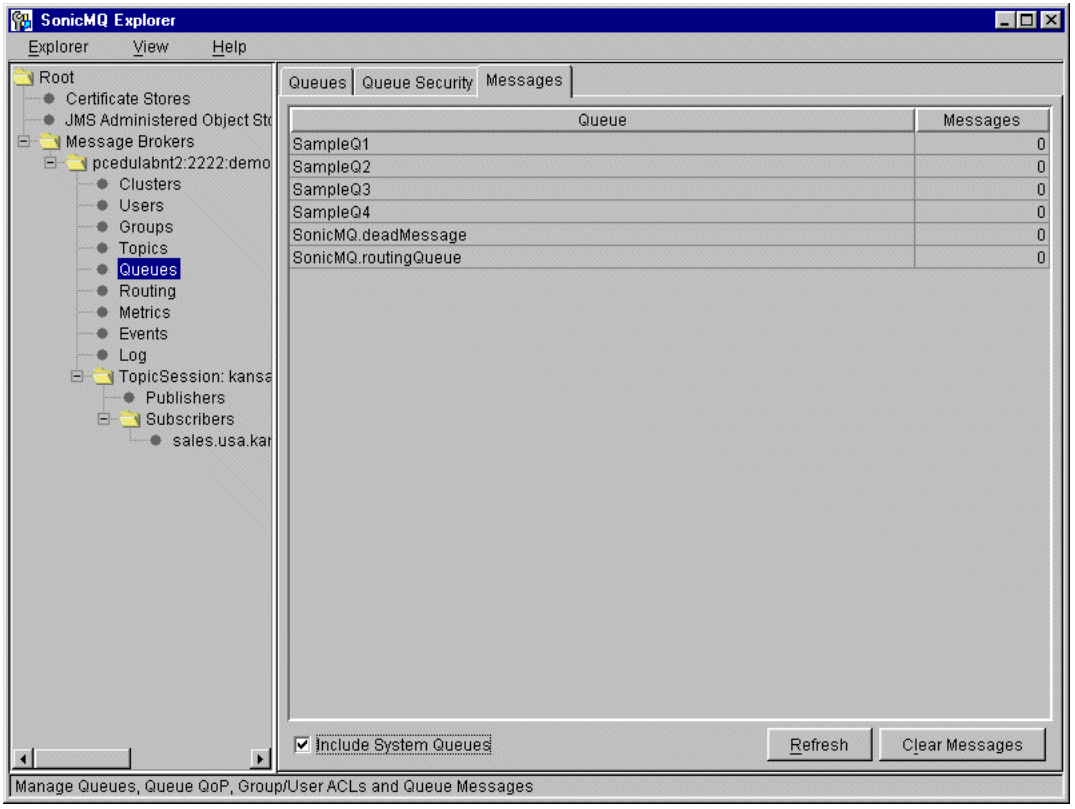


Figure 21. Queues: Messages

A table shows the queues in the left column and the number of messages currently stored in the queue in the right column. This number might include expired messages that have not yet been removed from the queue.

➤ To show system queues:

By default, SonicMQ does not show system (internal) queues.

- ☐ Select the **Include System Queues** check box to display the system queues, including the dead message queue.

➤ **To update the queue information:**

- ☐ Click **Refresh** to poll the server.

➤ **To clear the messages from a queue:**

1. Select the queue.
2. Click **Clear Messages**.

Routing

When you select the **Routing** node for a server in the tree panel, the right panel displays the tabs shown in Figure 22 under the following conditions:

- **Default Routing** — Available only for cluster configuration servers and stand-alone servers
- **Global Destinations** — Available for all servers

Default Routings Tab

The Default Routing tab lets you view existing routing table information and update existing routings.

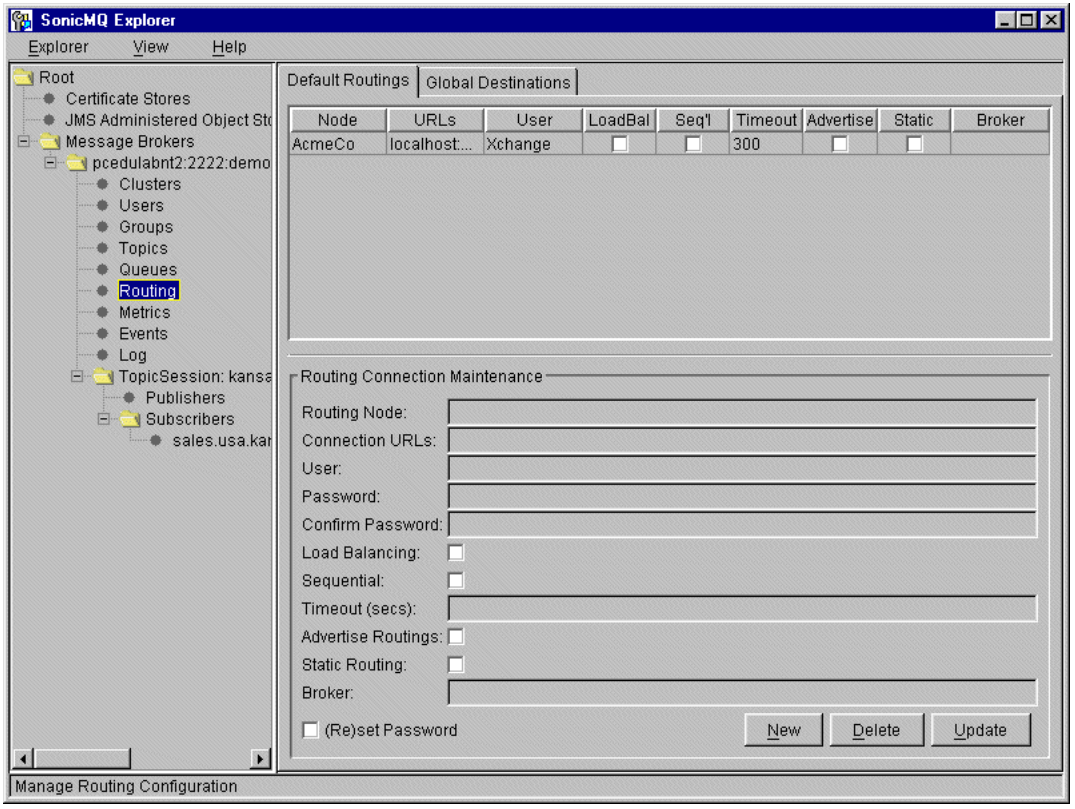


Figure 22. Routing: Default Routings

When you select a routing connection from the top panel, information about that connection appears in the lower **Routing Connection Maintenance** panel, where you can edit the parameters of the connection and select **Update** to apply your changes. Alternatively, you can create a new routing connection by supplying the date in the lower panel and selecting **New**. The routing data is displayed in the following UI elements:

- **Routing Node** — This field identifies the routing destination node to which the connection is made.
- **Connection URLs** — You must provide at least one routing connection path (message server connection URL). If you want to provide multiple connection paths, specify them as a comma-separated list.
- **User** –This field specifies the connection user. If your connection URLs use SSL, this can be the reserved word `AUTHENTICATED`.
- **Password** and **Confirm Password** — You must supply a connection user password unless the connection user is `AUTHENTICATED`.
- **Load Balancing** — If checked and the connection is to a routing node that is a cluster, the connection is made to a server in the cluster selected by a load-balancing algorithm.
- **Sequential** — If checked, the first connection attempt is made using the first connection URL. If unchecked, the first connection attempt is made using a URL randomly selected from the list of connection URLs.
- **Timeout** — This field specifies the length of time in seconds before an inactive routing connection is disconnected. A value of `0` (zero) represents an infinite timeout.
- **Advertise Routings** — If checked, information about all known global queues in the node is sent to the routing destination node for dynamic routing configuration purposes. If unchecked, this information is not sent.
- **Static Routing** — If checked, the routing connection is always used to route messages to the routing destination node. If unchecked, connections initiated from the routing destination node can be used for routing subsequent messages.
- **Broker** — For static routing connections, this field determines the server from which the routing connection is originated. For a nonclustered server, this defaults to the local server name.
- **(Re)set Password** — This check box is checked when you are entering new routing connection information, to indicate that you must set a password. If you are updating existing routing information, the check box is unchecked by default. To change the password, check this box and enter the new password in the **Password** field before clicking **Update**.

Global Destinations Tab

When you choose the **Global Destinations** tab, a list appears showing the **Routing Node** name and **Global Queue** name for each global destination currently known by the server, as shown in Figure 23.

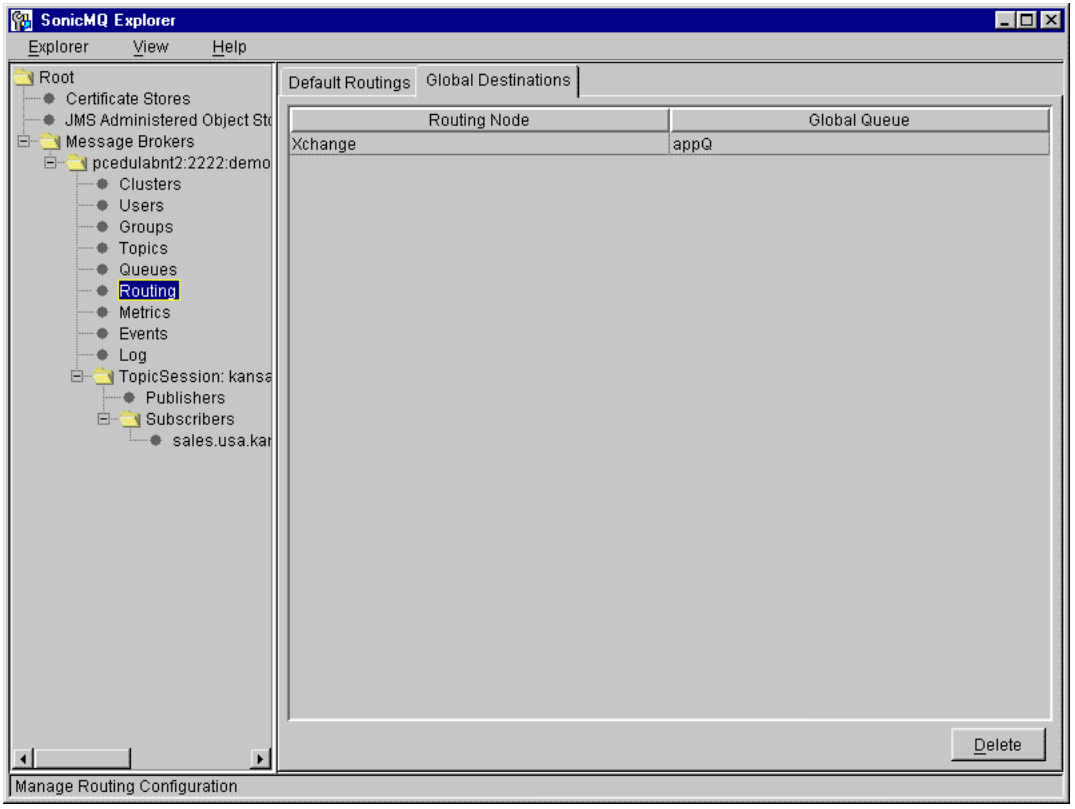


Figure 23. Routing: Global Destinations

If you delete a global destination, you remove the information about how to route messages from the selected server to another server.

➤ **To delete global destinations:**

1. Select one or more global destinations.
2. Choose **Delete**.

Metrics

When you select the **Metrics** node for a server in the tree panel, the right panel displays a **Summary** tab (as shown in Figure 24). The **Metrics** node also allows you to see **views**, which provide a dynamic graphical plot of a metric value over a selected time span.

Summary Tab

The Summary tab shows a list of metrics together with their current values.

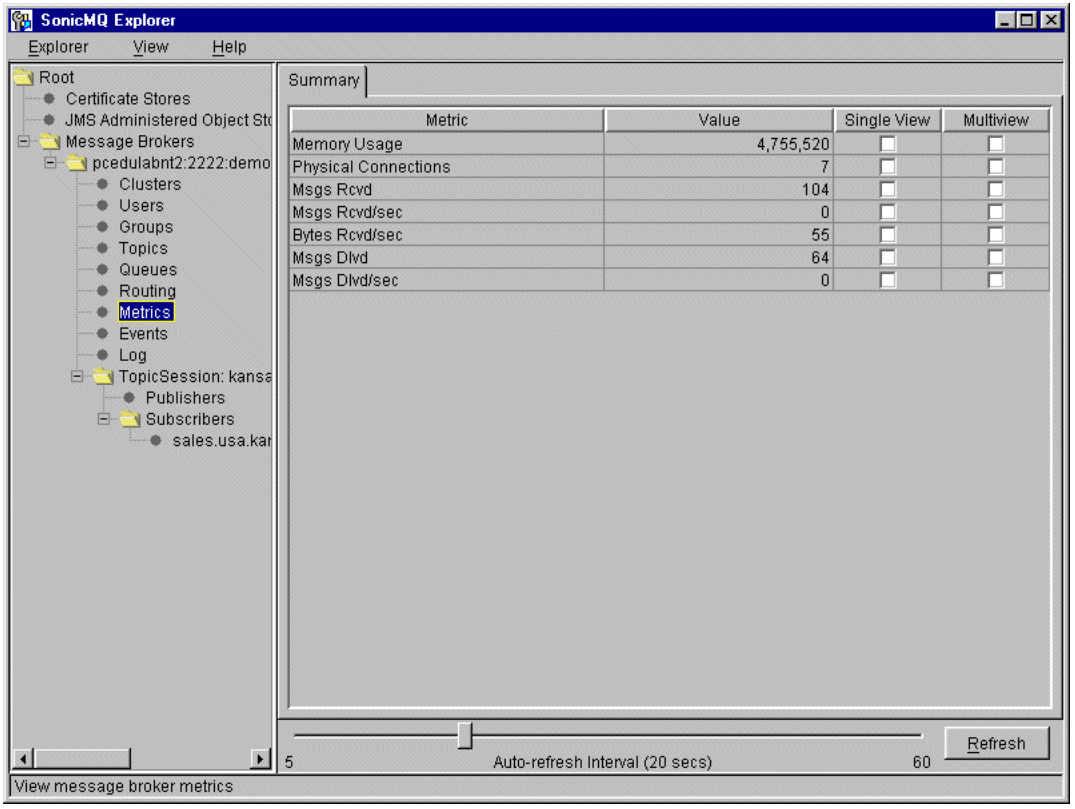


Figure 24. Metrics: Summary

The following metrics are provided:

- **Memory Usage** — The amount of memory (measured in bytes) in the JVM currently being used by the server
- **Physical Connections** — The current number of socket connections
- **Msgs Rcvd** — The number of messages received by the server from publishers or senders in the last interval.
- **Msgs Rcvd/sec** — The Msgs Rcvd metric as a per-second rate
- **Bytes Rcvd/sec** — The byte-count of the Msgs Rcvd as a per-second rate

- **Msgs Dlvd** — The number of messages delivered in the last interval, including messages delivered to subscribers in the Pub/Sub domain and messages delivered to queue receivers in the PTP domain
- **Msgs Dlvd/sec** — The Msgs Dlvd metric as a per-second rate

The server's default interval length for metrics collection is 10 minutes, with a refresh rate of 20 seconds. To change these values, see "METRICS_COLLECTION_INTERVAL" on page 61 and "METRICS_REFRESH_INTERVAL" on page 61.

The **Single View** and **Multiview** check boxes are selected by double-clicking. If you select one or more check boxes in the **Multiview** column, a **Multiview** tab appears. If you select check boxes in the **Single View** column, a single view tab appears corresponding to each checked metric, labeled with the name of the metric. The situation is illustrated in Figure 24.

- Click **Refresh** to get the latest metrics from the server.
- Periodically, the data is automatically refreshed. Adjust the **Auto-refresh Interval** slider to change the frequency with which current metric values are requested from the server.

Note You gain nothing by setting the auto-refresh interval smaller than the server's `METRIC_REFRESH_INTERVAL`, which by default is 20 seconds. If you need a smaller time interval, reduce this setting in the `broker.ini` file.

Multiview Tab

When you select the **Multiview** tab, the views that you checked on the **Summary** tab are displayed in a single panel, as illustrated in Figure 25.

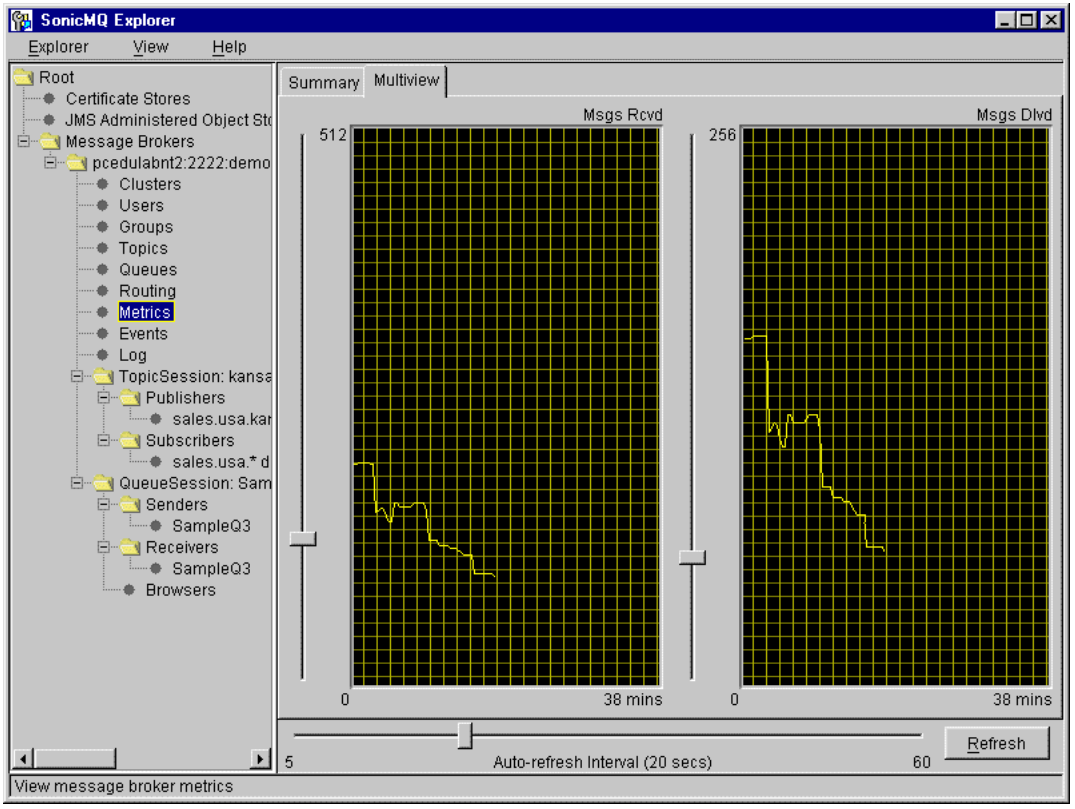


Figure 25. Metrics: Multiview

You can manipulate the metrics graphs in several ways:

- When you change the refresh interval by adjusting the **Auto-refresh Interval** slider, you also change the horizontal (time) scale of the view.
- You can change the maximum plotted time span by changing the time scale or by changing the size of the window to change the size of the graphs.
- You can change the vertical scale of each graph by moving the vertical slider to the left of the graph. The maximum value of the displayed quantity is always a power of two.

Single View Tab

When you select a **Single View** tab, such as **Memory Usage** in Figure 26, you see a tab that looks much like the **Multiview** tab, except that only one view for the specified metric is chosen. You can manipulate the view on a **Single View** tab in the same way as the **Multiview** tab views.

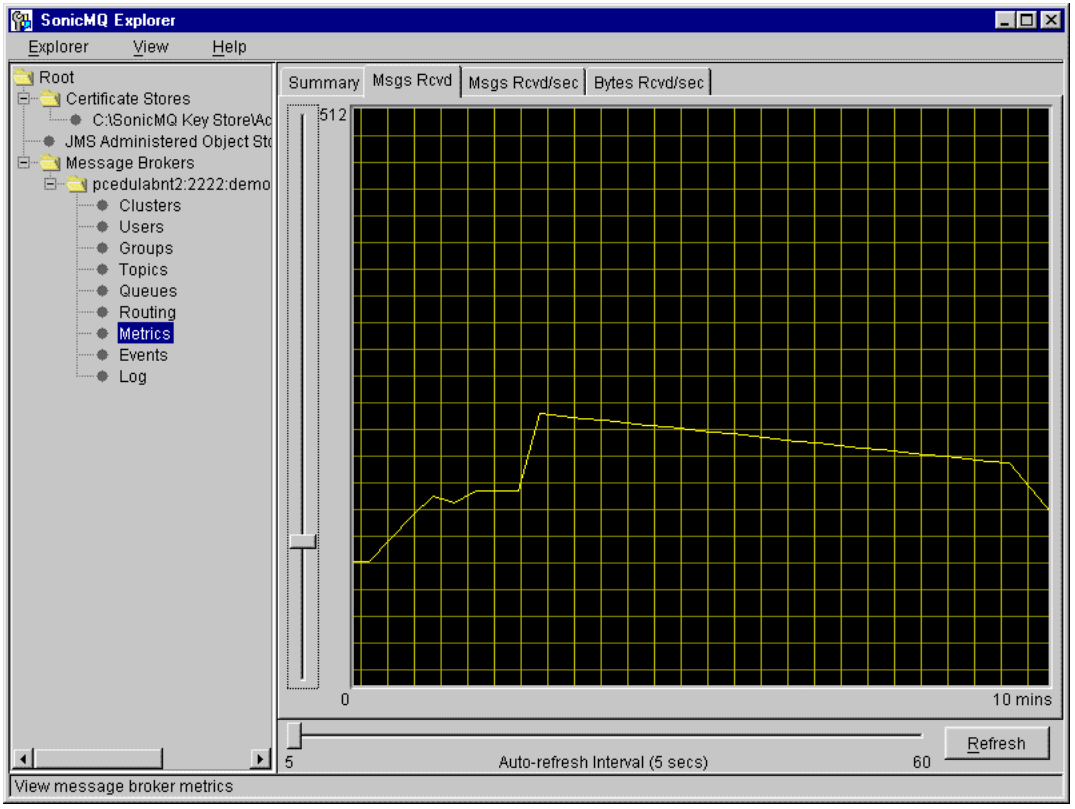


Figure 26. Metrics: Memory Usage

Events

When you select the **Events** node for a server, the right panel shows a table of trackable server events, as shown in Figure 27.

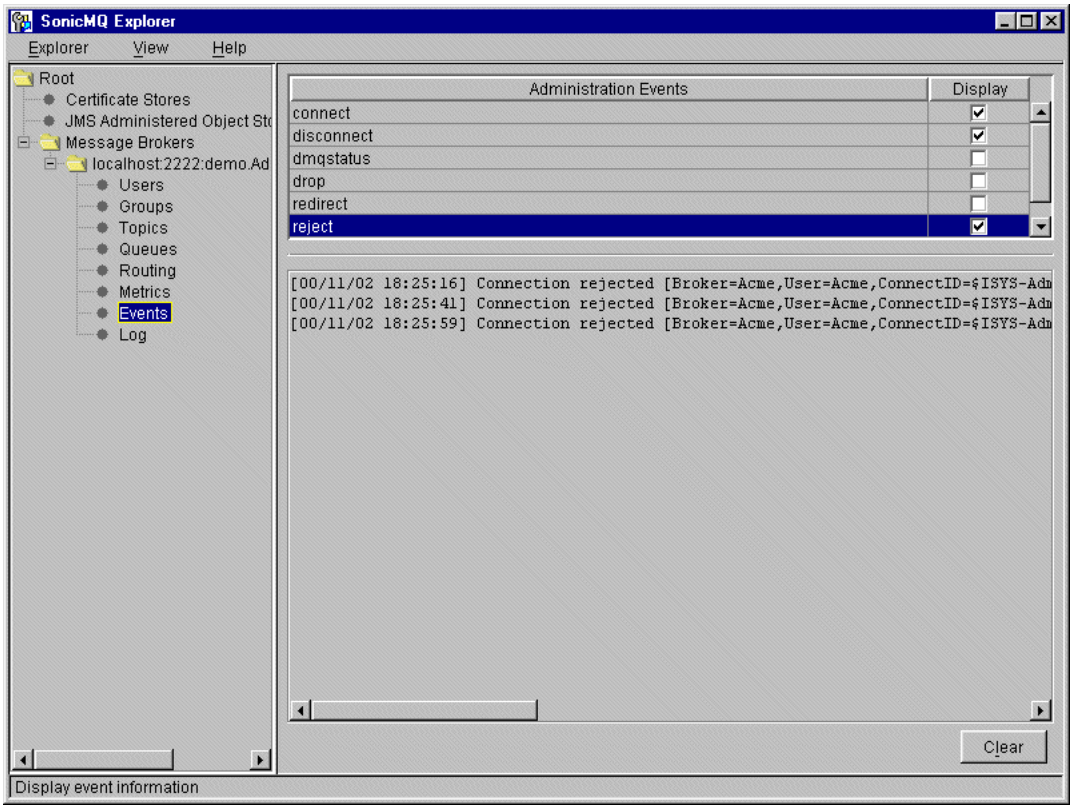


Figure 27. Events

The **Events List** in the lower panel can display the following event types:

- **connect** — Generated every time a connection is opened
- **disconnect** — Generated every time a connection is closed
- **dmqstatus** — Generated every time a message is added to the DMQ if the DMQ has exceeded a user-specified fraction of its maximum size
- **drop** — Generated every time a connection is lost without the client being disconnected, for example, if the client dies

- **redirect** — Generated every time a client connect attempt is redirected to another server in a SonicMQ cluster because of load balancing
- **reject** — Generated every time a connection is rejected
- **undelivered** — Generated every time a message is placed in the dead message queue (DMQ)

Events of a given type are displayed if the **Display** check box for that event type is checked. For example, Figure 27 displays three instances of the **reject** event generated by attempts to connect to a broker without the proper authentication.

➤ **To check or clear the box for a given event type:**

☐ Double-click the check box.

The display typically shows a maximum of 500 events, on a FIFO basis. To change this number, see "Customizing Explorer Behavior" on page 106.

If you click **Clear**, the **Events List** panel becomes blank.

Log

When you select the **Log** node for a server in the tree panel, the right panel changes to display the contents of the log for the selected server in a scrollable window, as shown in Figure 28:

- Click **Refresh** to get the latest copy of the server log.
- Click **Truncate** to clear the server log.

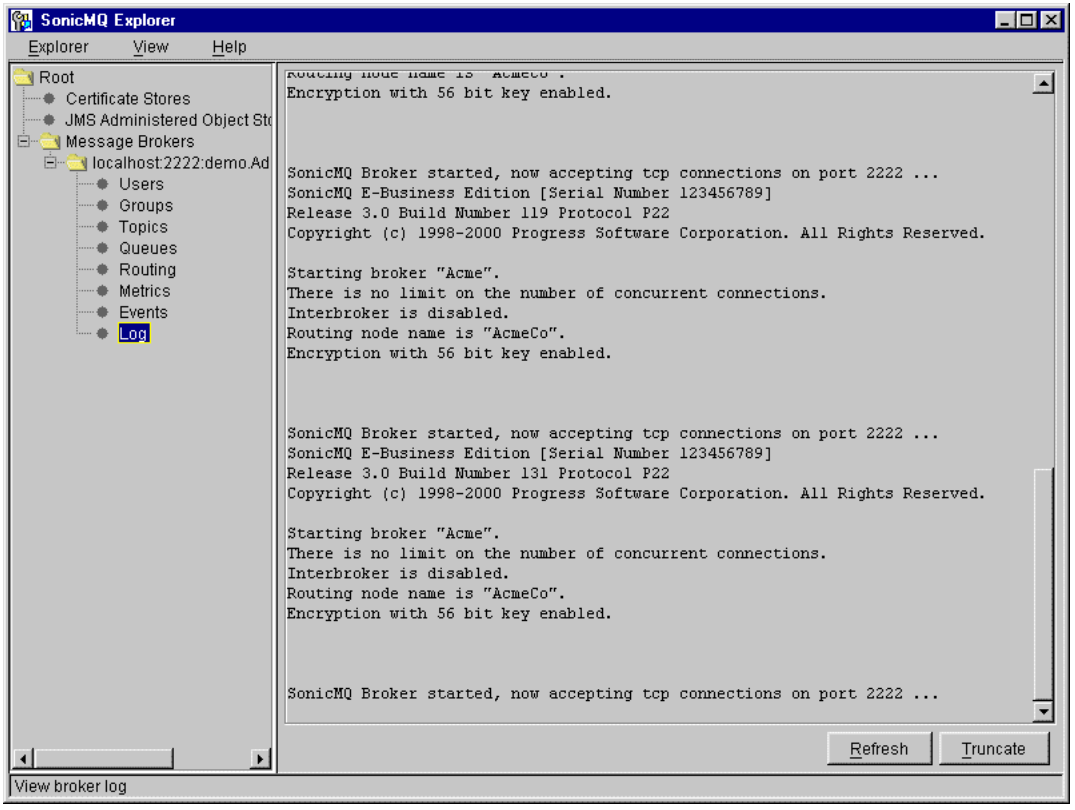


Figure 28. Log

Sessions and Connections

When you select a server in the tree panel, the right panel displays a **Sessions** tab and a **Connections** tab.

Sessions Tab

The **Sessions** tab lets you view established sessions and create new test sessions, as shown in Figure 29.

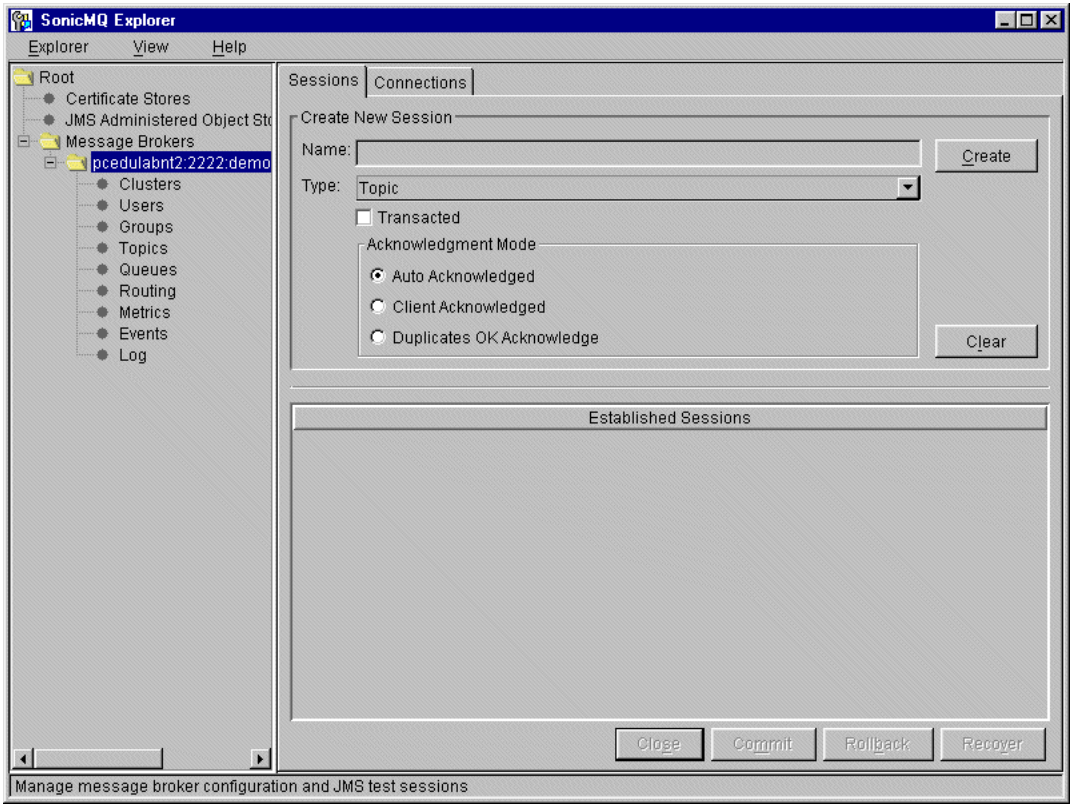


Figure 29. New Session

When creating a new session you have a number of options:

- You must specify a **Name** for the session.
- You must select the **Type** of session, **Topic** (for a TopicSession in the Publish and Subscribe domain), or **Queue** (for a QueueSession in the Point-to-Point domain).

- You might select **Transacted** to use transactions (see *Getting Started with SonicMQ* for an explanation). If **Transacted** is selected, **Client Acknowledged** and **Duplicates OK Acknowledge** are disabled, and the acknowledgement mode is set to **Auto Acknowledged**.
- If you do not select Transacted you must select an **Acknowledgement Mode**—**Auto Acknowledged**, **Client Acknowledged**, or **Duplicates OK Acknowledge** (see *Getting Started with SonicMQ* for an explanation).

Buttons at the bottom of the right panel allow you to:

- Close the session.
- **Commit** or **Rollback** a transaction (these buttons are active only if the session is **Transacted**).
- **Recover**, which causes all unacknowledged messages received to be redelivered. This has no effect in **Auto Acknowledged** or **Duplicates OK Acknowledge** modes.

If the session type is **Queue**, you can use the newly established session to send messages, receive messages, or both.

If the session type is **Topic**, you can use the newly established session to publish messages, to subscribe to messages, or both.

Connections Tab

When you select the connections tab a list of active connections to the server is shown, as in Figure 30.

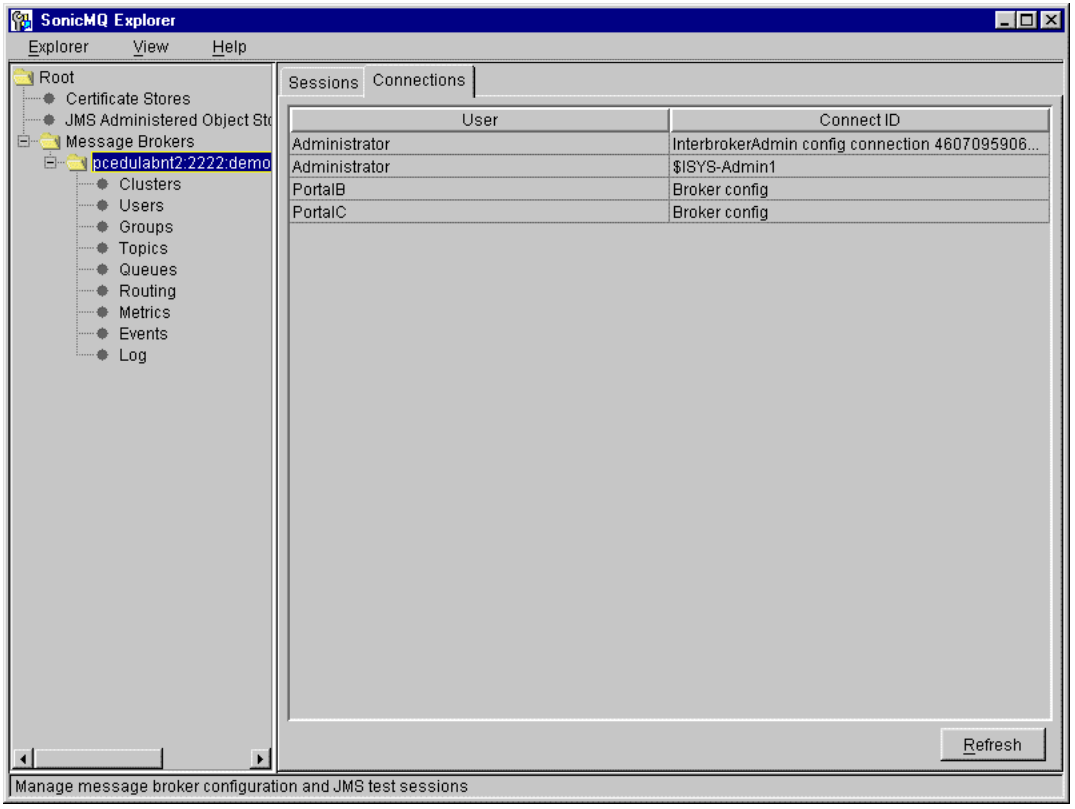


Figure 30. Connections

When you click **Refresh**, the list is updated with the currently active connections.

Creating a Test Publish and Subscribe Session

SonicMQ Explorer gives you a powerful way to simulate parts of your application and to demonstrate the behavior of various server modes of behavior by establishing a test Publish and Subscribe session.

To publish (or send) messages, a connection must be started. The connection starts automatically when there are adequate resources:

- Server
- Connection to the server
- Session on the connection
- Message mechanism (publisher, subscriber, listener, receiver, sender)

As you add, modify, or delete any resources, the connection automatically stops to allow the update and then restarts to allow publishing or sending of messages.

➤ To establish a test Publish and Subscribe session:

1. Select a server connection node in the tree panel. (See Figure 29 on page 151.)
2. Establish a session:
   - **2.1** Type any unique string in the **Name** field and press **Enter**.
   - **2.2** Select **Topic** from the **Type** drop-down list.
   - **2.3** Check **Transacted** if you require a transacted session.
   - **2.4** Select the **Auto Acknowledged** radio button in the **Acknowledgement Mode** group.
   - **2.5** Click **Create**.

   The session appears in the tree panel with **Publishers** and **Subscribers** nodes.

3. If you click on the **Publishers** node in the tree panel, the right panel lets you view established publishers (if any) and create new publishers, as shown in Figure 31.

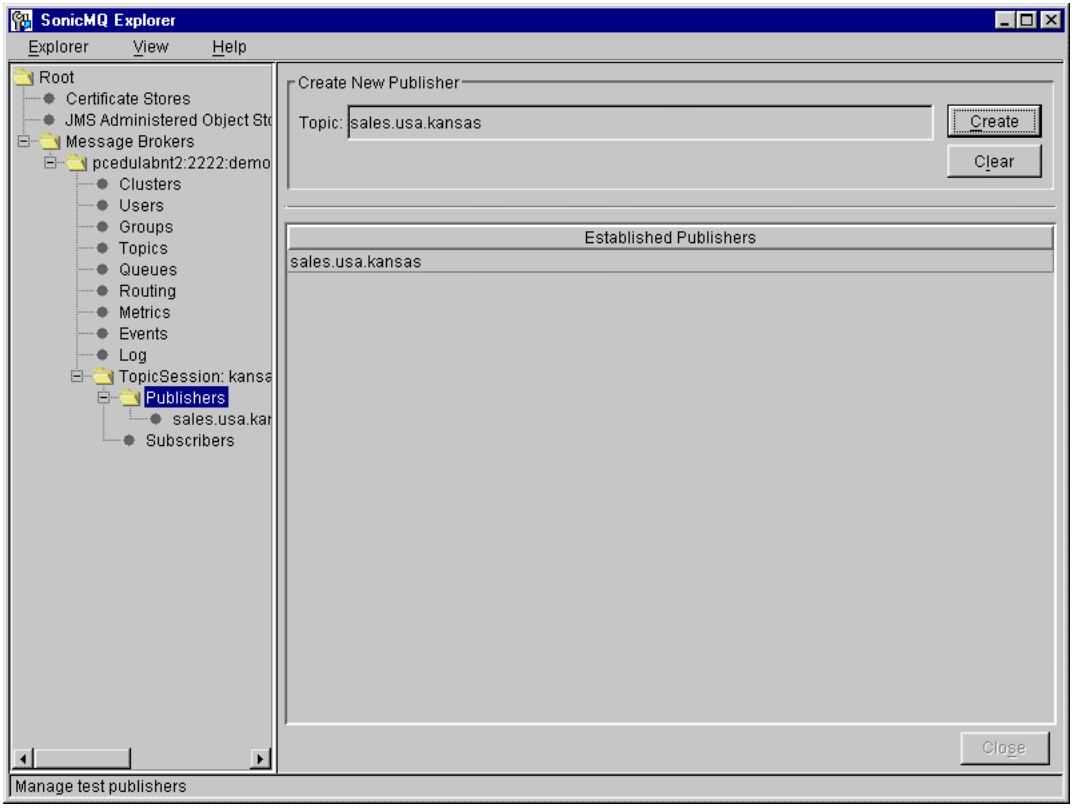


Figure 31. Publishers

4. To create a new publisher, enter the name of the topic where messages are to be published in the **Topic** field and click **Create**. Do not use the asterisk (*), pound (#), backslash (\), or dollar sign ($) characters.
5. Click on the **Subscribers** node in the tree panel. The right panel lets you view established subscribers (if any) and create new subscribers, as shown

in Figure 32, where the publisher **sales.usa.kansas** and the subscriber **sales.usa.*** have been created:

- The **Message Selector** field enables you to enter query values based on JMS header fields and properties to filter out undesired messages. The syntax of this string is based on a subset of the SQL-92 conditional expression syntax. For details, see the JMS Specification, *Java Messaging Service, Version 1.0.2*.
- If you check **Durable** and fill in the **Name** field, you can create a durable subscription with the given name.
- If you check **No Local Delivery**, the subscriber does not receive messages from publishers on the same connection.

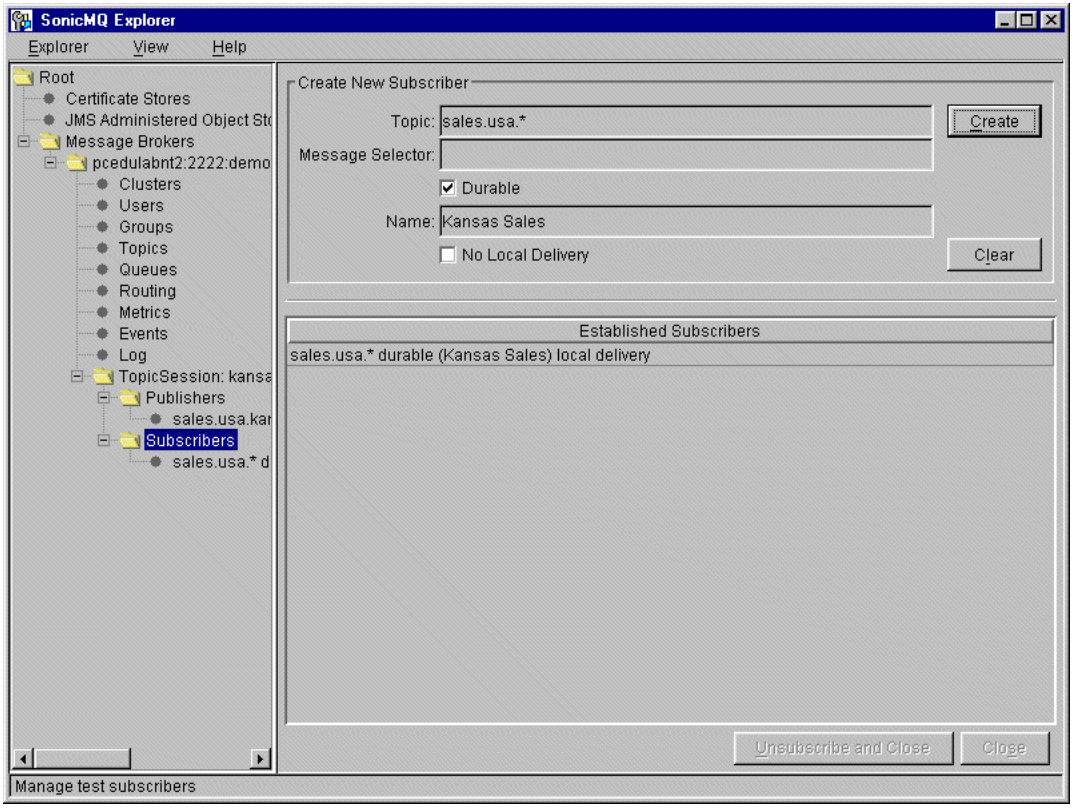


Figure 32. Subscribers

6. Click on the **Subscribers** node.
7. Fill in the **Topic** field and click **Create** to create new subscribers to a topic. The asterisk (*) and pound sign (#) can be used as template characters. See the "Hierarchical Name Spaces" chapter in the *SonicMQ Programming Guide* for more information.

   With publishers and subscribers established, you can publish and subscribe to messages.
8. Select the **Message Brokers** node in the tree panel and select a server from the server list in the lower panel.
9. Select a publisher in the tree panel. The right panel displays three tabs: **Header**, **Properties**, and **Body**, as shown in Figure 33. The names on the tabs refer to the three parts of a message.

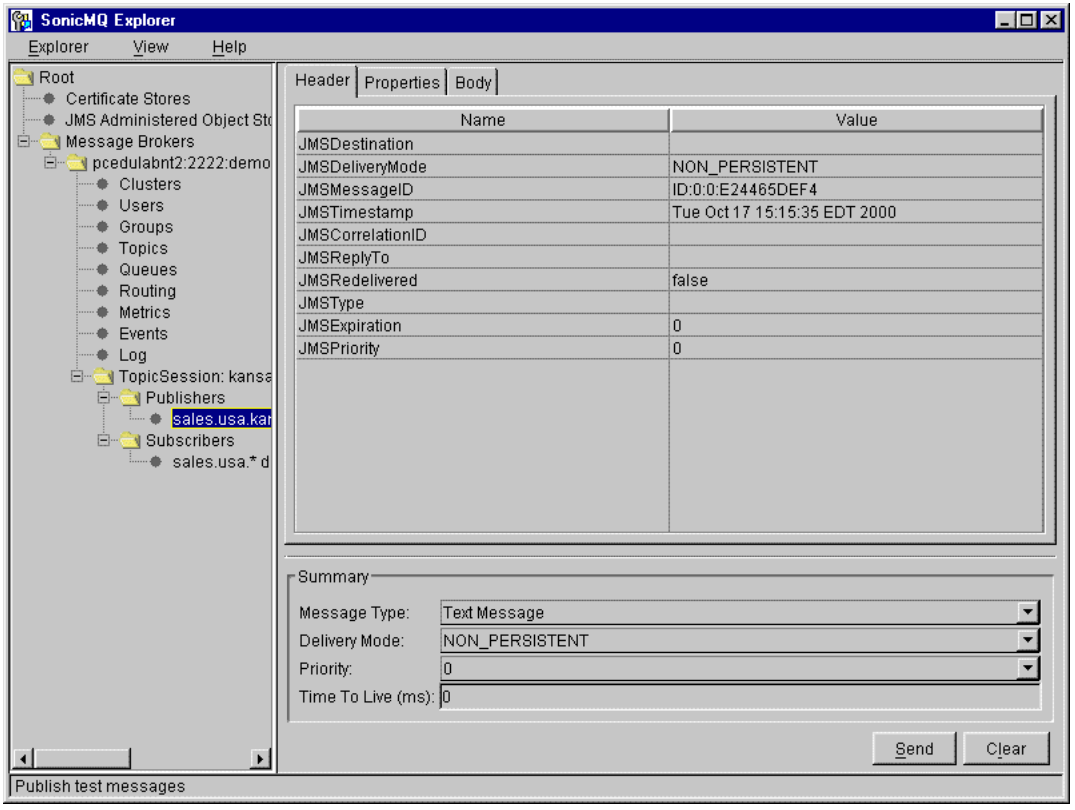


Figure 33. Message Header

You can edit only the following items in the header list:

- **JMSCorrelationID**
- **JMSReplyTo**
- **JMSType**

In the Summary section, you can specify:

- **Message Type**, `Message`, `Text Message`, or `XML Message`
- **Delivery Mode**, `NON_PERSISTENT` or `PERSISTENT`
- **Priority**, with `0` (zero) the lowest and `9` the highest
- **Time To Live,** (in milliseconds) with `0` (zero) indicating no expiration

See the "Messages" section in the *SonicMQ Programming Guide* for more information.

10. Select the **Properties** tab, which allows you to define property values specified in the JMS specification, as shown in Figure 34. These include the following SonicMQ-specific properties:

    - JMS_SonicMQ_preserveUndelivered
    - JMS_SonicMQ_notifyUndelivered

    For more information on these properties, see the messages chapter in the *SonicMQ Programming Handbook*.

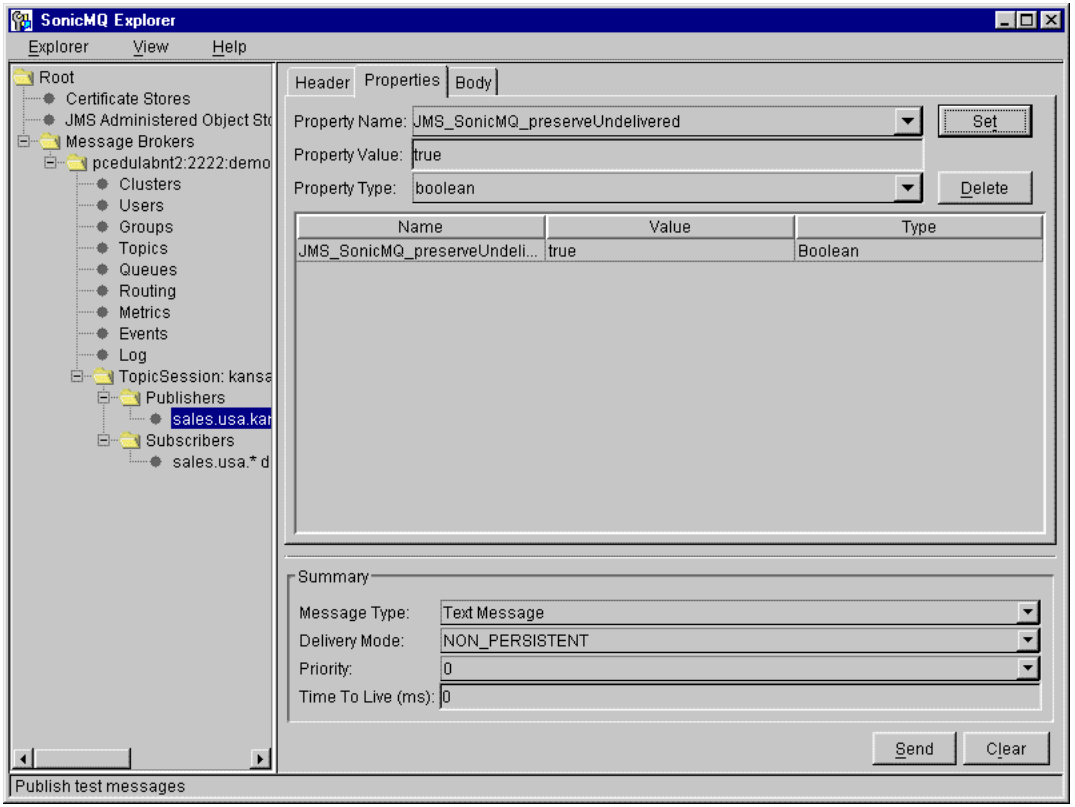


Figure 34. Message Properties

11. Select the **Body** tab to compose the body of the message, as shown in Figure 35.

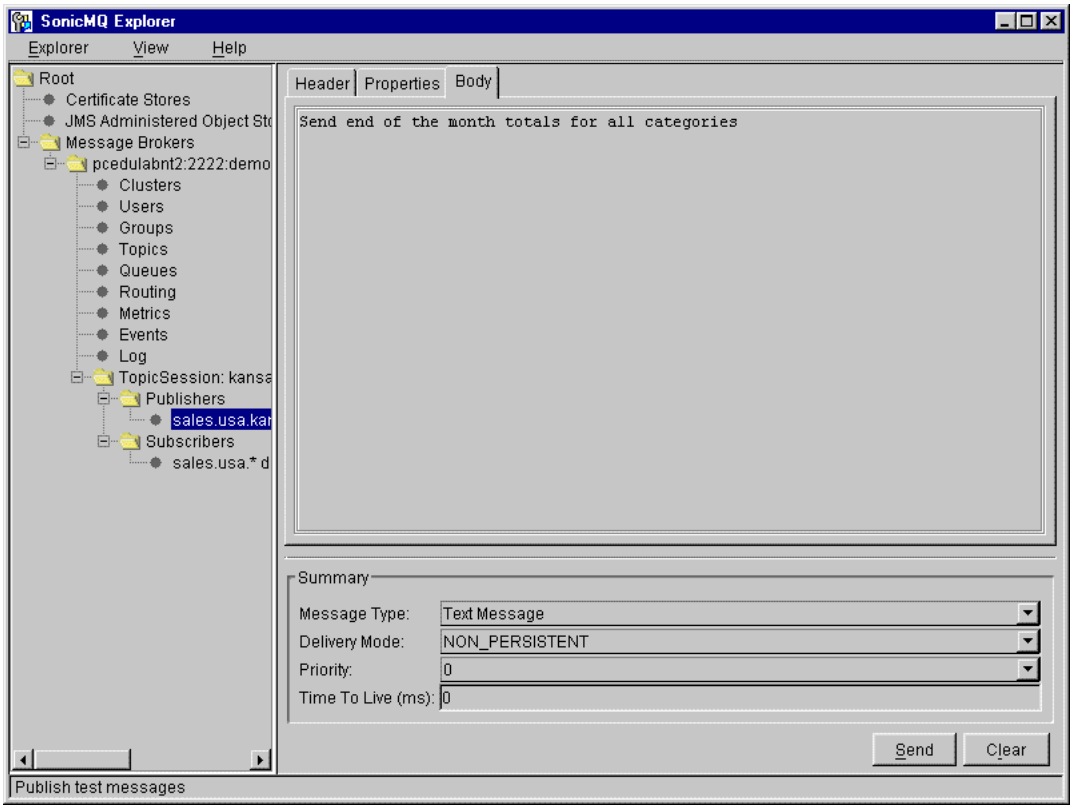


Figure 35. Message Body

12. Click **Send** to send your message.

13. If you created an appropriate subscriber, select a node under **Subscribers** corresponding to the topic under which you published your message. The window shown in Figure 36 allows you to view the messages delivered to this subscriber. Select the message. The same message you sent in Step 12 appears in the bottom panel, as shown in Figure 36 on page 161.

Note The message was published to the topic **sales.usa.kansas**, and therefore received by the subscriber **sales.usa.***.

Note By default, the number of messages held in the **Subscribed Messages** table is 50. You can change this number by setting the Explorer startup property `admin.maxMsgs`.

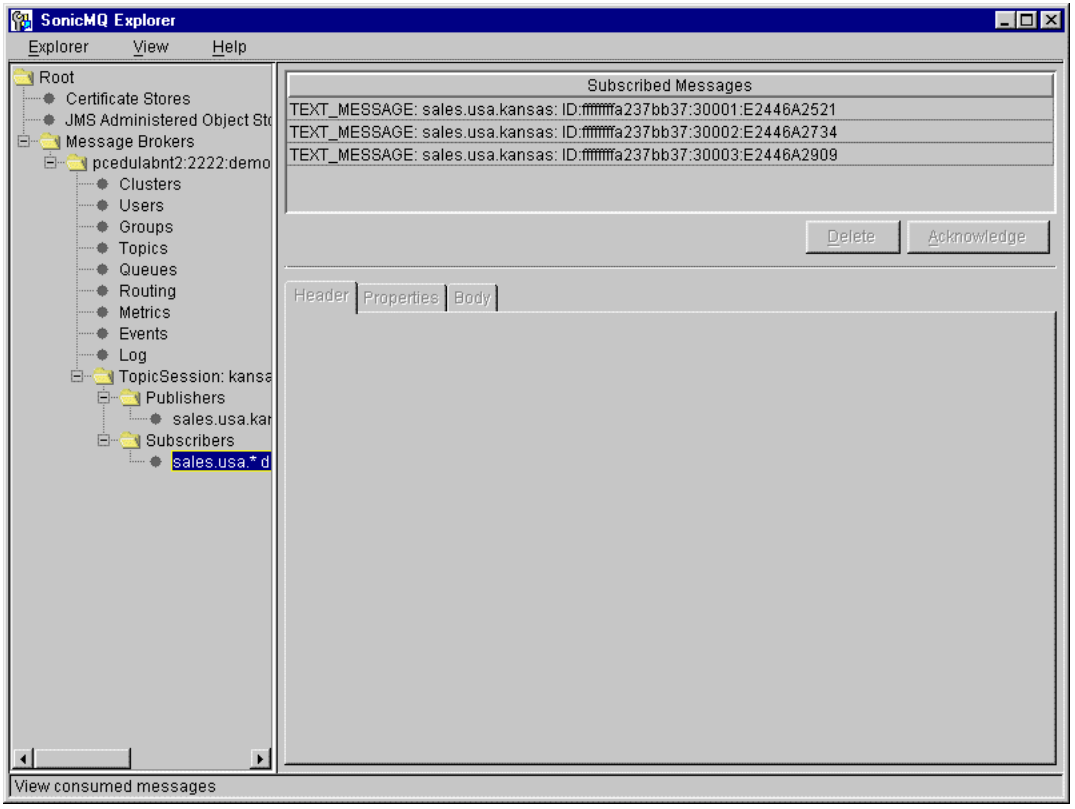


Figure 36. Subscribed Messages

14. In this window, you can perform the following operations on messages:

    - To delete one or more messages without explicitly acknowledging them, select the messages and click **Delete**.
    - To acknowledge one or more messages, select the messages and click **Acknowledge**. An acknowledgment is sent back to the server. This is only effective if the session was established in **Client Acknowledged** mode. (Messages can also be automatically acknowledged, depending

on how the session was established. See "Sessions and Connections" on page 150 for details.)

Creating a Test Point-to-Point Session

SonicMQ Explorer gives you a powerful way to simulate parts of your application and to demonstrate the behavior of various server modes of behavior by establishing a test Point-to-Point session.

➤ **To establish a test Point-to-Point session:**

1. Select a connection node in the tree panel. (See Figure 29 on page 151.)
2. Establish a session:
   - 2.1 Type any unique string in the **Name** field and press **Enter**. (You can even leave the field blank.)
   - 2.2 Select **Queue** from the **Type** drop-down list.
   - 2.3 Check **Transacted** if you require a transacted session.
   - 2.4 Select the **Auto Acknowledged** radio button in the **Acknowledgement Mode** group.
   - 2.5 Click **Create**.

   The session appears in the tree panel with **Senders**, **Receivers**, and **Browsers** nodes.

3. If you click on the **Senders** node in the tree panel, the right panel lets you view established senders that have been started from this Explorer session (if any) and create new senders, as shown in Figure 37.

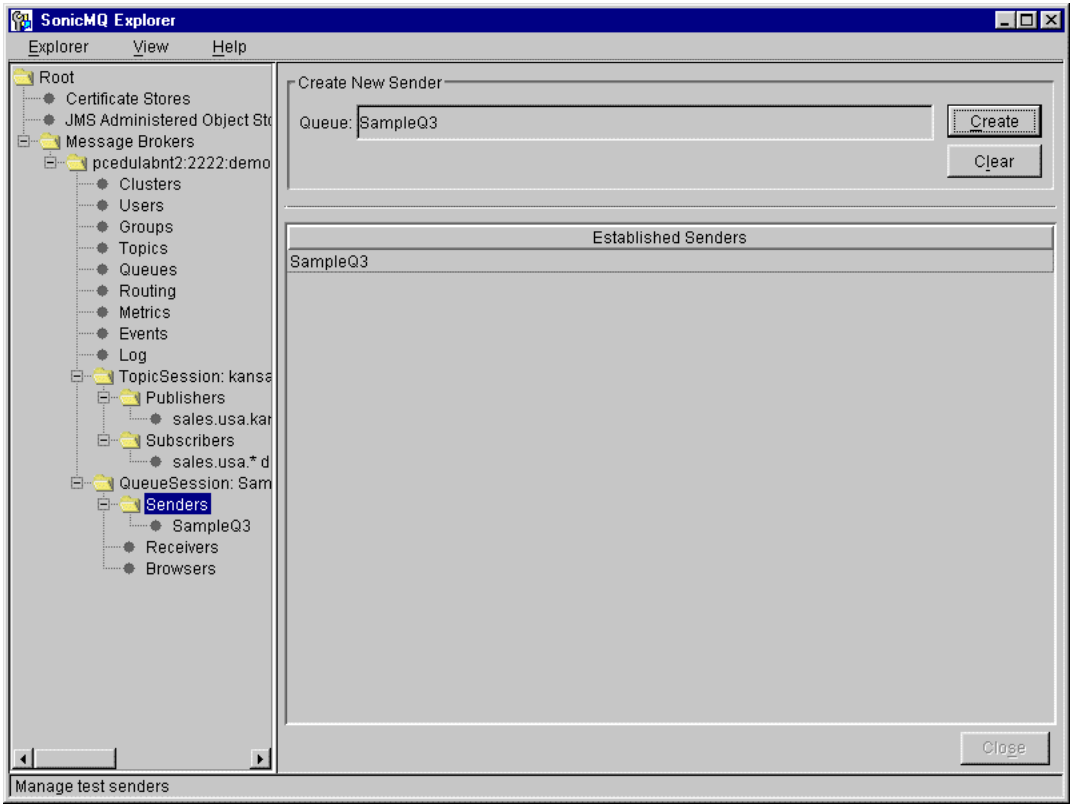


Figure 37. Senders

4. To create a new sender, enter the name of the queue you wish to send to in the **Queue** field and click **Create**. The queue must already exist. If the queue you want to send to does not exist, go to the **Queues** node to create it.

5. If you click on the **Receivers** node in the tree panel, the right panel lets you view the receivers that have been established in this session (if any) and create new receivers, as shown in Figure 38.

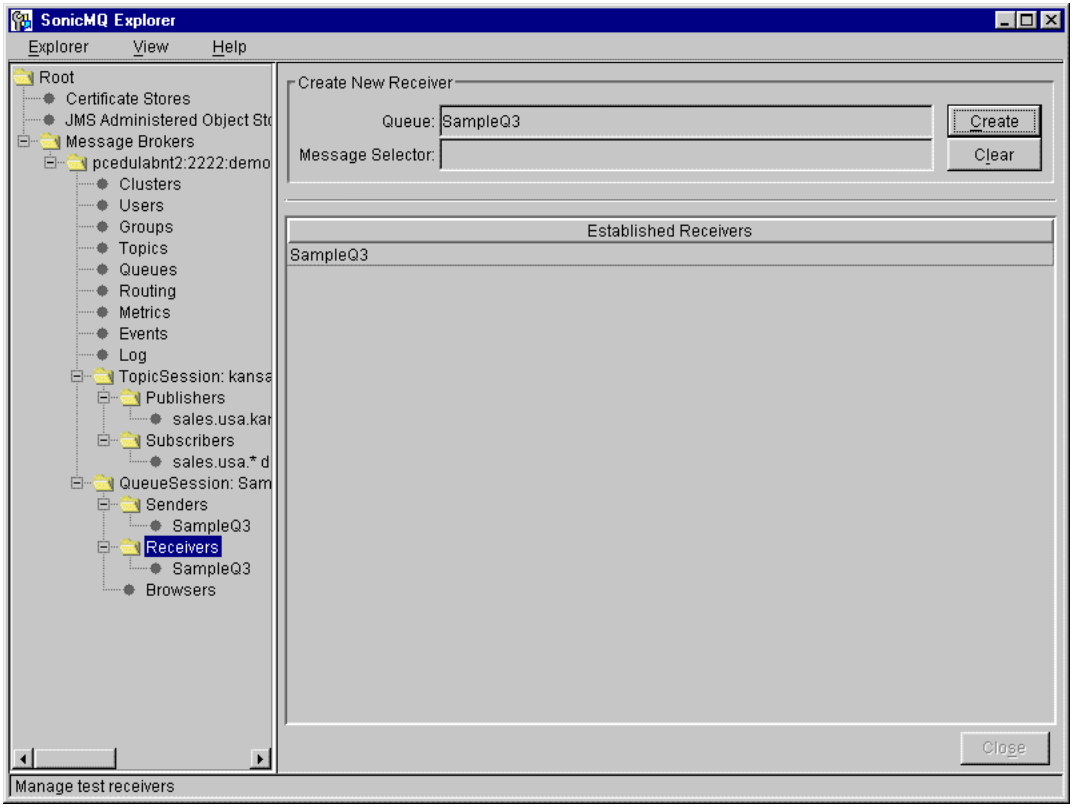


Figure 38. Receivers

6. Click on the **Receivers** node.
7. Fill the **Queue** field with the name of an existing queue. If the queue you wish to send to does not exist, go to the **Queues** node to create it.
8. If desired, fill in the **Message Selector** field with a query based on header fields and properties to filter out undesired messages. The syntax of this string is based on a subset of the SQL-92 conditional expression syntax. For details, see the JMS Specification, *Java Messaging Service, Version 1.0.2*.

9. Click **Create** to create a new receiver.
10. To create additional receivers, click **Clear** and then repeat Step 7 through Step 9.

    With senders and receivers established, you can send and receive messages.
11. Select the **Message Brokers** node in the tree panel and select a server from the server list in the lower panel.
12. Select a sender in the tree panel. The right panel displays three tabs: **Header**, **Properties**, and **Body**. The panel is identical to the one that appears when you select a publisher in the Publish and Subscribe model. (See Figure 33 on page 157.)
13. Fill in the **Header**, **Properties**, and **Body** tabs for your message. (See Step 9 on page 157 through Step 11 on page 160 in "Creating a Test Publish and Subscribe Session" for details.)
14. Click **Send** to send your message.
15. Assuming you have created an appropriate receiver, select a node under **Receivers** for the queue to which you sent your message. The window shown in Figure 39 allows you to view the messages sent to this receiver. Select the message. The same message you sent in Step 14 appears in the bottom panel.

Note By default, the number of messages held in the **Subscribed Messages** table is 50. You can change this number by setting the Explorer startup property `admin.maxMsgs`.

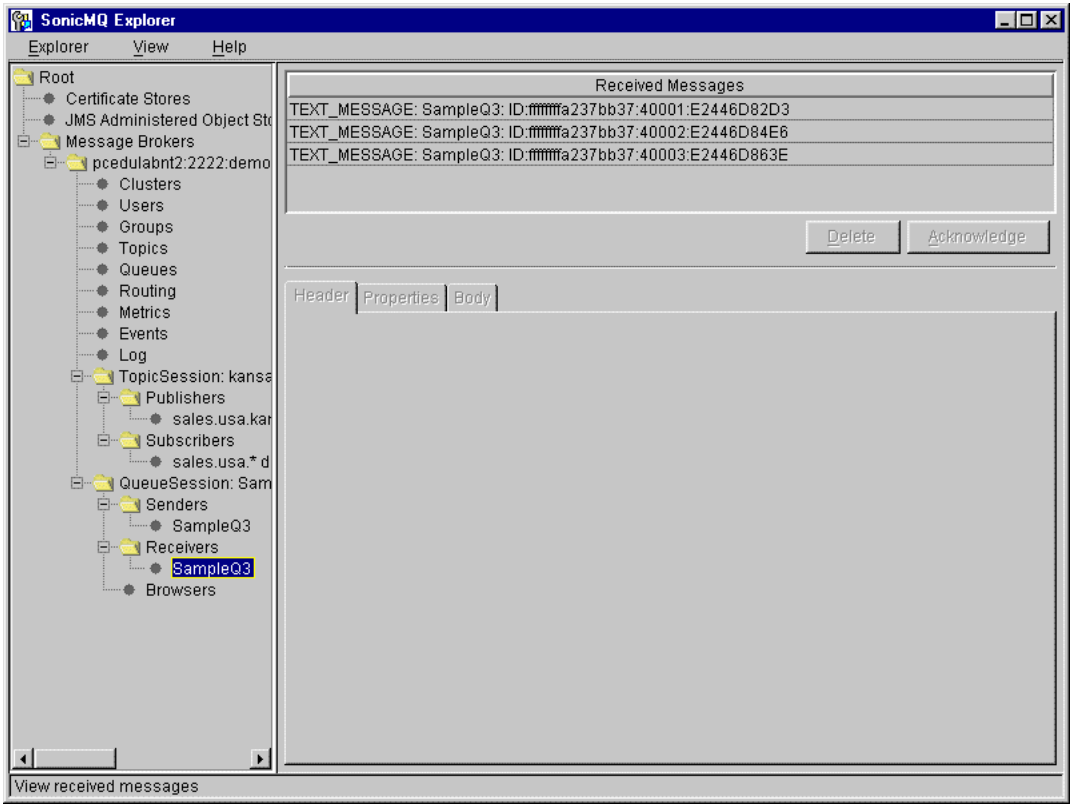


Figure 39. Received Messages

16. In this window, you can perform the following operations on messages:

    - To delete one or more messages without acknowledging them, select the messages and click **Delete**.
    - To explicitly acknowledge one or more messages, select the messages and click **Acknowledge**. An acknowledgment is sent back to the server. This is only effective if the session was established in **Client Acknowledged** mode. (Messages can also be automatically acknowledged, depending on how the session was established. See "Sessions and Connections" on page 150 for details.)

17. Click the **Browsers** node to browse messages in the specified queues, as shown in Figure 40.

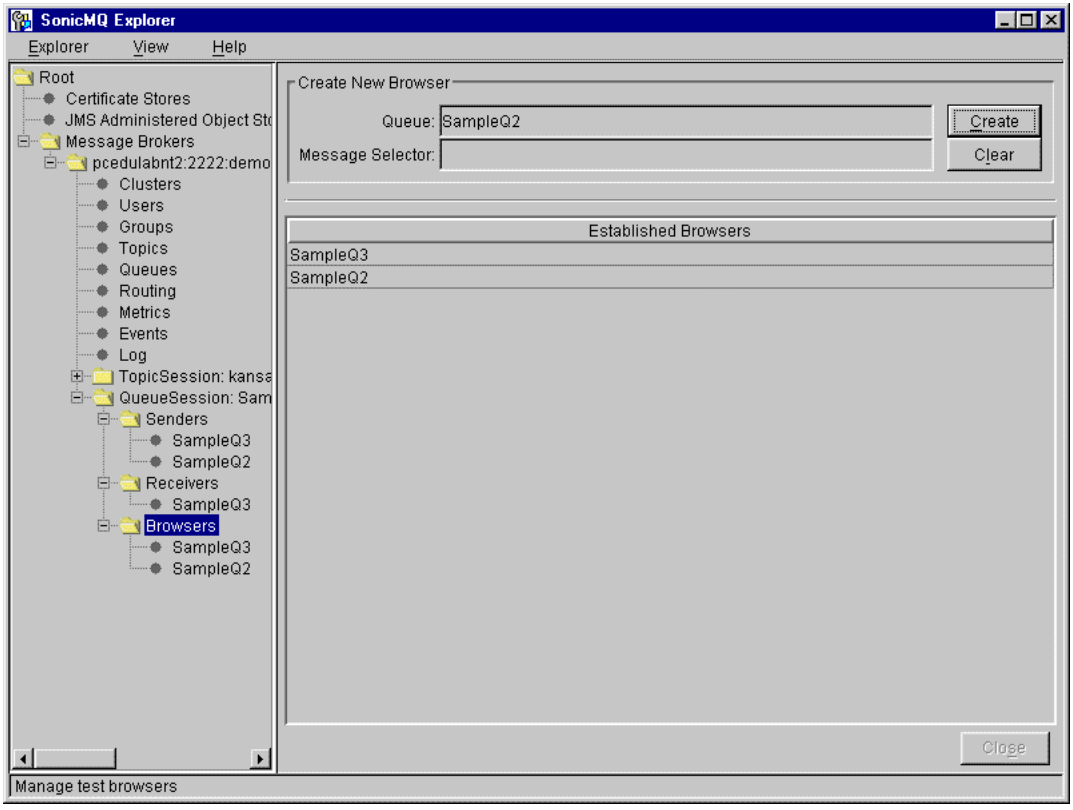


Figure 40. Queue Browser Creation

18. To create a queue browser, type the name of an existing queue that you want to browse and click **Create**. The new queue browser appears in the Established Browsers list.

19. Select the queue browser for the queue you want to view, and click the appropriate position-selector button (black arrow heads) to position the browser cursor to the messages you want to see, as shown in Figure 41.

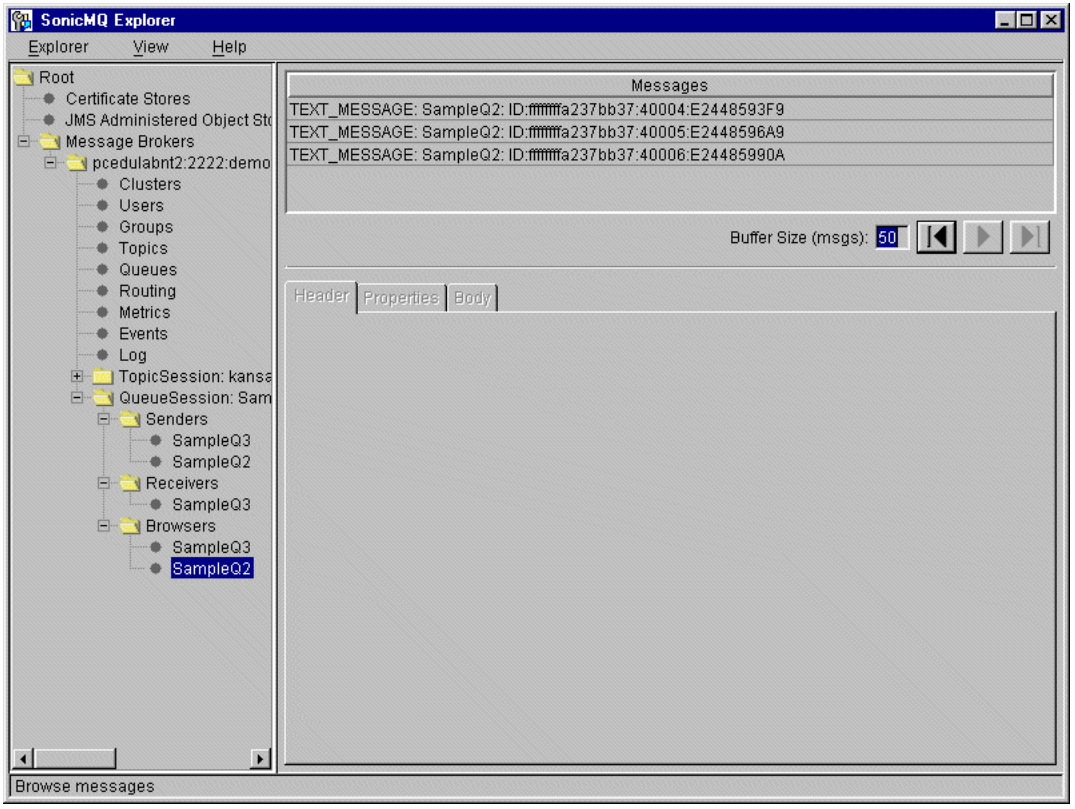


Figure 41. Queue Browser

Note To start browsing messages, you must first choose the left-most position-selector button. You can restart the browse by choosing the same button when the browser cursor is at any position in the queue. The two right-most position-selector buttons are active only when there are more messages on the queue than the specified buffer size. The move-forward position-selector button shows the next buffer-size number of messages in the queue. The move-to-end position-selector button shows the last buffer-size number of messages in the queue.

JMS Administered Object Stores

When you select **JMS Administered Object Stores** in the tree panel, the right panel changes as in Figure 42.

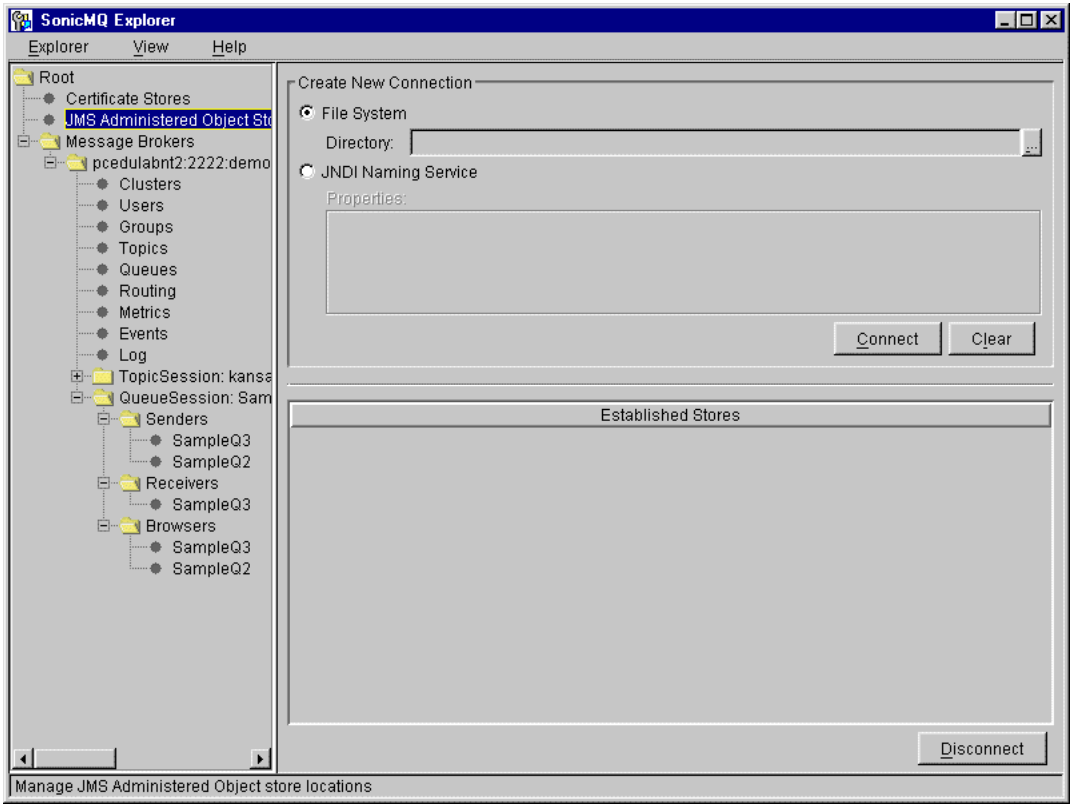


Figure 42. JMS Administered Object Stores

The way you connect to a JMS administered object store depends on whether you want to use a file-based JMS administered object store or an existing JNDI-based store.

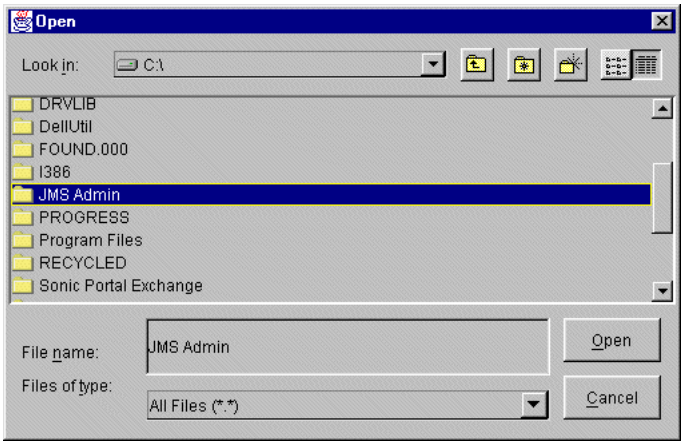
Note To use a JNDI-based store, you must place the `.jar` files for a JNDI provider on the Explorer CLASSPATH. If you do not have these files on the Explorer CLASSPATH, Explorer displays an Exception window with a message that begins, **Cannot instantiate class...**, for the particular JNDI service you are trying to connect.

File-based Object Stores

A file-based object store maintains JMS administered objects in a file that you specify in your file system.

➤ **To use a file-based JMS administered object store:**

1. Select **File System**.
2. Specify a directory by typing a directory name in the **Directory** field. Alternatively, click the **Browse** button at the right side of the **Directory** field to see the folder selection dialog box:



To choose a nondefault directory:

2.1 Click on the drop-down list at the top of the dialog box to see a list of available drives.

2.2 Click on a drive to see a list of root-level directories. Directories are listed in alphabetical order, but with all capitalized names preceding all lowercase names.

2.3 Repeatedly double-click on directories to navigate downwards through the directory tree.

2.4 When you reach the desired directory, click once to select the directory and then click **Open**.

3. Click **Connect**.

This opens a store in the specified directory, as shown in Figure 43. In this case, JMS administered objects that you create get serialized to files in that directory.

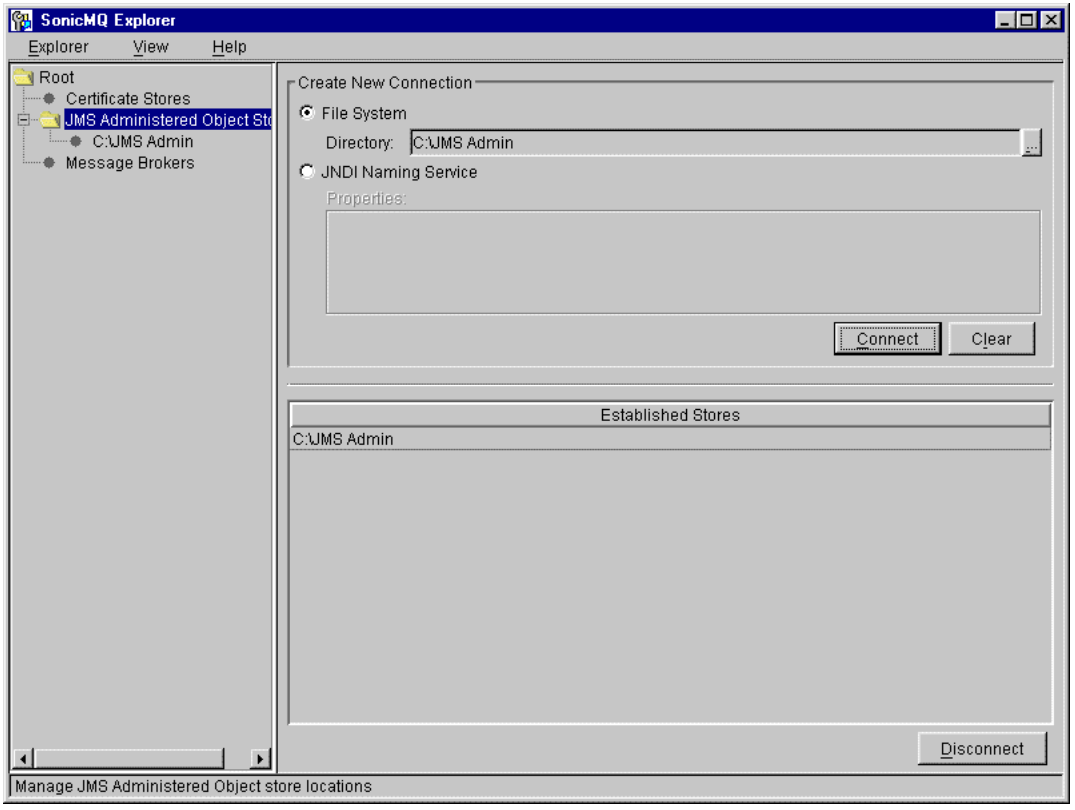


Figure 43. File-based Object Store

JNDI-based Object Stores

A JNDI-based object store maintains JMS administered objects in a specified JNDI naming service.

➤ **To use a JNDI-based JMS administered object store:**

1. Select **JNDI Naming Service**.
2. Provide JNDI initialization as property=value pairs in the **Properties** text box. The specific pairs depend on the JNDI store being used. At a minimum, the initial context must be specified. For example, you might

enter the following in the **Properties** text box (as a single line, with *no line breaks*):

```
java.naming.provider.url="ldap://diablo:389/ou=JMSAO,ou=SonicMQ,o=
bedford.progress.com", java.naming.factory.initial=com.sun.jndi.ld
ap.LdapCtxFactory
```

You can specify default JNDI connection properties in the script that brings up Explorer, `explorer.bat` (Windows) or `explorer.sh` (UNIX or Linux). You can then provide additional values or overrides in the text box.

3. Click **Connect**.

This opens a JNDI-based store as shown in Figure 44. In this case, JMS administered objects that you create get stored to the specified naming service.

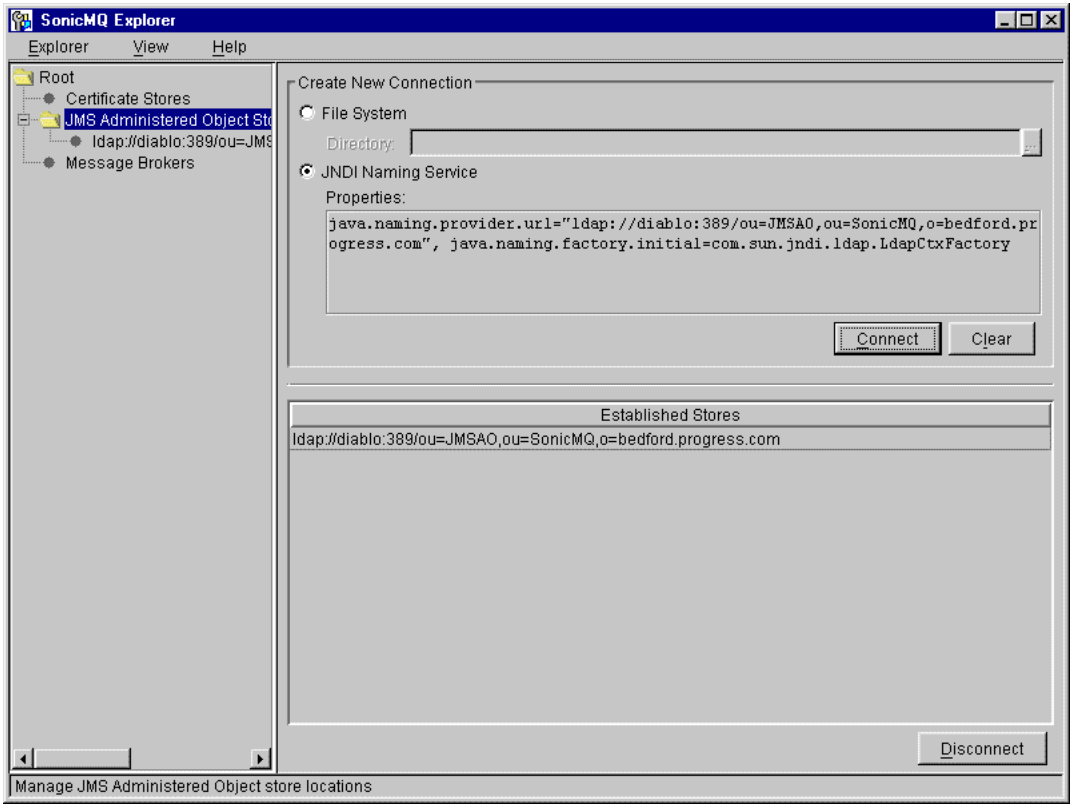


Figure 44. JNDI-based Object Store

Using JMS Administered Stores

After an object store has been established, the object store appears in the tree panel. When you select a JMS administered object store in the tree panel, the right panel is redrawn with two tabs, **Destinations** and **Connection Factories**.

Destinations Tab

The **Destinations** tab is selected by default, as shown in Figure 45. A Destination is either a Topic or a Queue.

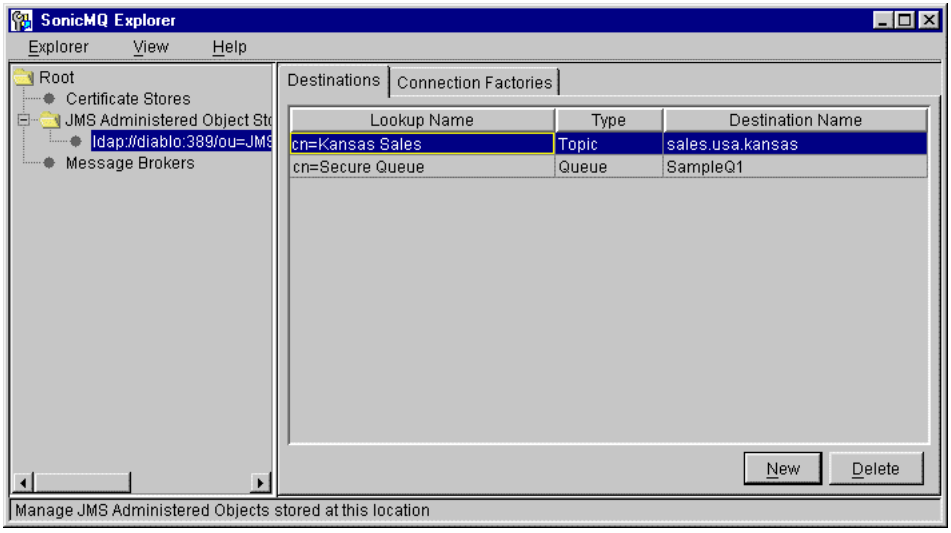


Figure 45. Destinations

If Destination and ConnectionFactory objects already exist in the store, they are displayed in their respective tabs.

➤ **To add a Destination to a JMS administered object store:**

1. Click **New**.
2. Enter the **Lookup Name** of your choice. The name of a destination in a file-based store is used to generate the filename for the object. The name of a destination in a JNDI/LDAP store might be `"cn=Kansas Sales"`.
3. Select the Destination **Type** from the drop-down list. The choices are **Queue** and **Topic**.

4. Enter the **Destination Name**, the name of the Queue or Topic, as it appears in Explorer or the Admin tool, and press **ENTER**.

➤ To delete a Destination from a JMS administered object store:

1. Select the Destination in the **Destinations** table.
2. Click **Delete**.

Connection Factories Tab

If you choose the **Connection Factories** tab, the panel changes, as shown in Figure 46.

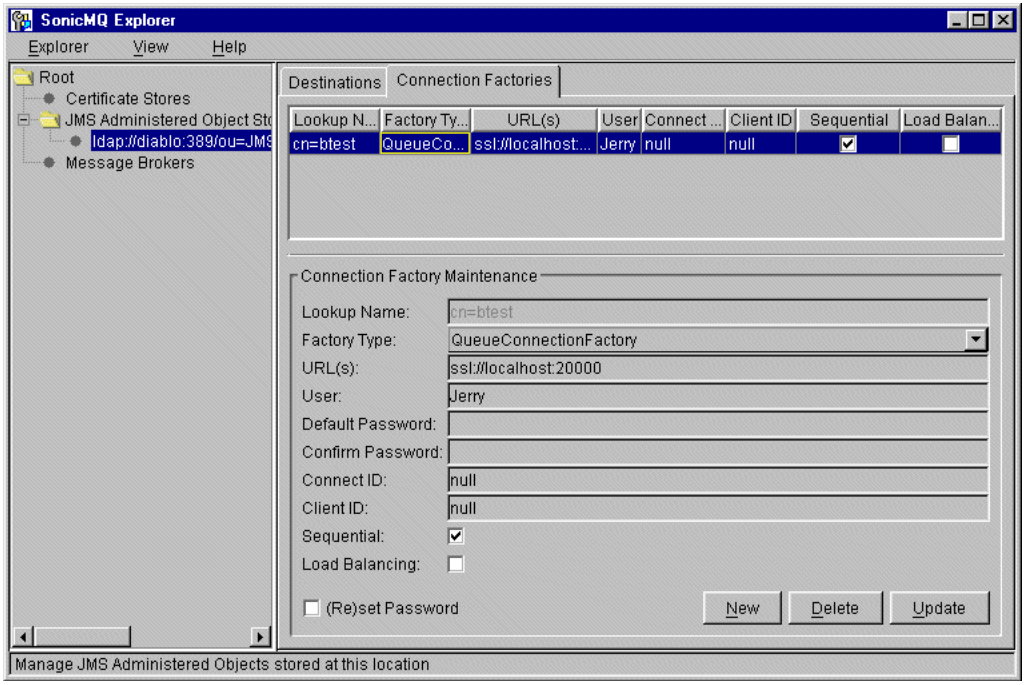


Figure 46. Connection Factories

The top panel displays a table of existing connection factories. The bottom panel, **Connection Factory Maintenance**, contains a number of fields that specify the properties for the connection factory selected in the table:

- **Lookup Name** — A name for the connection factory object. The name of an object in a file system-based store is a filename (without the `.sjo` extension), while the name of an object in a JNDI/LDAP store might be `"cn=Accounting"`.
- **Factory Type** — A drop-down list specifying the connection type from the following choices:
  - **QueueConnectionFactory**
  - **TopicConnectionFactory**
- **URL(s)** — A comma-delimited list of one or more connection URLs. If the list contains more than one connection URL. Each connection URL is of the form {**`tcp://`**|`ssl://`|`http://`}*`host`*{**`:2506`**|`:`*`port`*} where:
  - *`host`* is a resolvable IP name or address.
  - *`port`* is the port number for the connection.
  - `tcp`, `ssl`, and `http` are possible connection protocols. SSL is only supported for SonicMQ Professional Developer and E-Business Editions.
- **User** — The default user.
- **Default Password** and **Confirm Password** — The default user's password.
- **Connect ID** — A connection identifier, used to name a JMS client connection to the server in a queue or topic session. By setting **Connect ID** to `null`, you let a single client establish multiple simultaneous connections to the server. By setting **Connect ID** to any other value, you let the specified client establish only a single simultaneous connection.
- **Client ID** — The JMS client identifier. The value `null` means the user can set **Client ID** at connection time, through a `setClientID()` method on the connection. If the factory has a **Client ID** other than `null`, it cannot be changed by the client using the ensuing connection.
- **Sequential** — If checked, the first URL in **URL(s)** is tried first. Otherwise, random selection determines which URL is tried first.

- **Load Balancing** — If checked, the requested connection can be redirected for load balancing. Otherwise, the requested connection cannot be redirected.
- **(Re)set Password** — If checked, the value in **Default Password** is confirmed and stored as the new password for **User**.

➤ **To add a connection factory to a JMS administered object store:**

1. Click **New**.
2. Enter the **Lookup Name** of your choice.
3. Select the connection factory type from the **Factory Type** drop-down list.
4. Enter one or more URLs in the **URL(s)** field as the URL or list of URLs to use for the connection.
5. If desired, enter a default username in the **User** field.
6. If desired, enter a default user password in the **Default Password** field. As you enter the password, each character you type is echoed as an asterisk (*). You then must fill in the **Confirm Password** field with the same password.
7. Enter a convenient identifier in the **Connect ID** field.
8. Enter the client identifier in the **Client ID** field.
9. Uncheck the **Sequential** check box if you want the listed connection URLs to be tried in random order.
10. Check the **Load Balancing** check box if you want the requested connection to be redirected for load balancing.
11. Check the **(Re)set Password** check box to store the specified password.
12. Click **Update**.

➤ **To change a connection factory from a JMS administered object store:**

1. Select the connection factory in the **Connection Factories** table.
2. Change the fields and check boxes as you require in the **Connection Factory Maintenance** panel.
3. Click **Update**.

➤ **To delete a connection factory from a JMS administered object store:**

1. Select the connection factory in the **Connection Factories** table.
2. Click **Delete**.

Certificate Management Tool

When you select the **Certificate Stores** node in the tree panel of Explorer you access the Certificate Management Tool. See "Certificate Management" on page 104 for an overview of this tool.

Note To use this tool, you must place the `.jar` files for an SSL provider on the Explorer CLASSPATH. You can either use the JSAFE `.jar` files included in the `lib` folder of the SonicMQ install (`sslj.jar`, `certj.jar`, and `jsafe.jar`), or you can use `.jar` files that you must obtain from IAIK (for example, `iaik_jce_full.jar`). If you do not have these files on the Explorer CLASSPATH, Explorer displays an Exception window with the message, **Unable to locate a KeyStore implementation class file**, when you attempt to access the Certificate Management Tool.

Certificate Stores Panel

When you select the **Certificate Stores** node, the right panel of the Explorer main window allows you to open certificate stores, as shown in Figure 47.

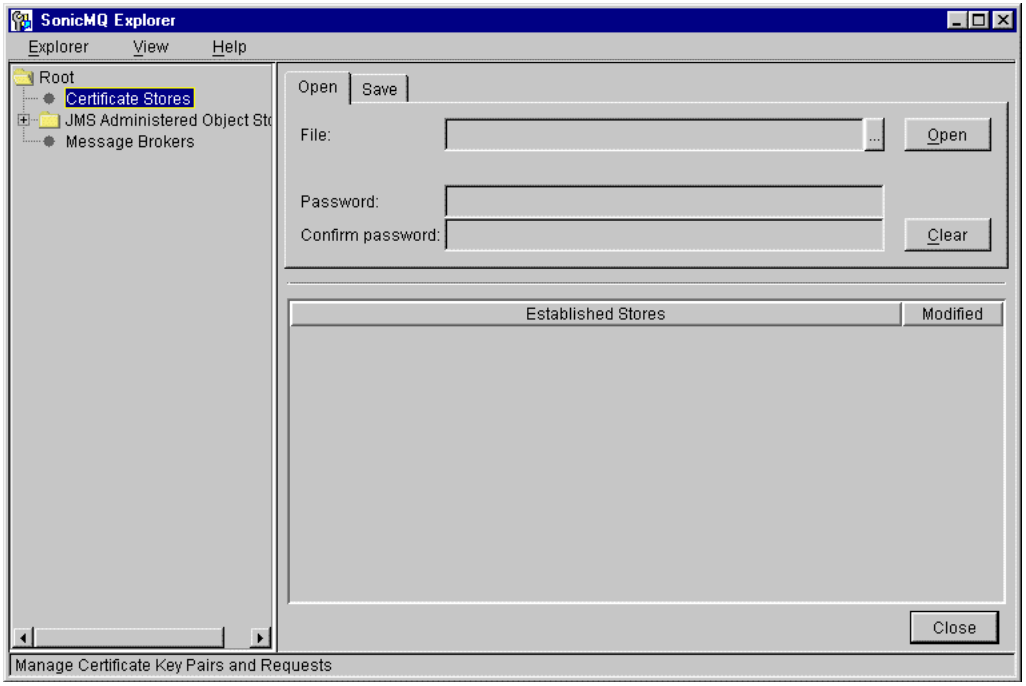


Figure 47. Certificate Stores

The Certificate Store is where where the certificates, keys, and related data are stored. This data is serialized, DES encrypted, and stored as a file on the file system.

Important The Certificate Store file (`KeyStore`) holds information that is critical to SonicMQ's SSL communication protocol. Be sure to back up this file frequently to protect against file system corruption.

The **Certificate Stores** panel lists established Certificate Stores opened in Explorer.

Open Tab

The **Open** tab allows you to create or open an existing Certificate Store.

➤ **To open a Certificate Store:**

1. To create a new store, enter the name of the Certificate Store **File**, or use the browse button to locate an existing Certificate Store. You can protect a new Certificate Store or open an existing Certificate Store protected by a password by entering and confirming the **Password**.
2. Once you have provided the file location and password, click the **Open** button to connect the Explorer to the Certificate Store. This displays the store established, as shown in Figure 48.

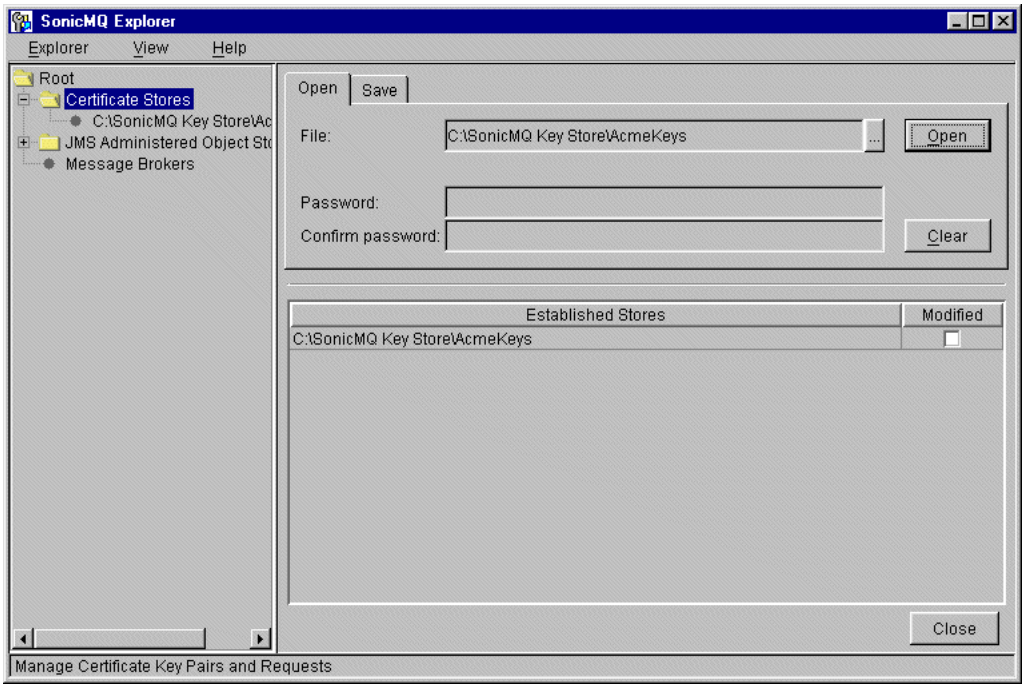


Figure 48. Opening Certificate Stores

Save Tab

The **Save** tab allows you to save modifications to an open Certificate Store.

➤ To save a Certificate Store that you have modified:

1. Select an established store in the list that has its **Modified** check box activated
2. Click the **Save** tab.
3. If you want to set or change the password before saving, activate the **(Re)set Password** check box and enter or clear the password, as you want.
4. Click the **Save** button.

This saves the Certificate Store and clears the **Modified** check box, as shown in Figure 49.

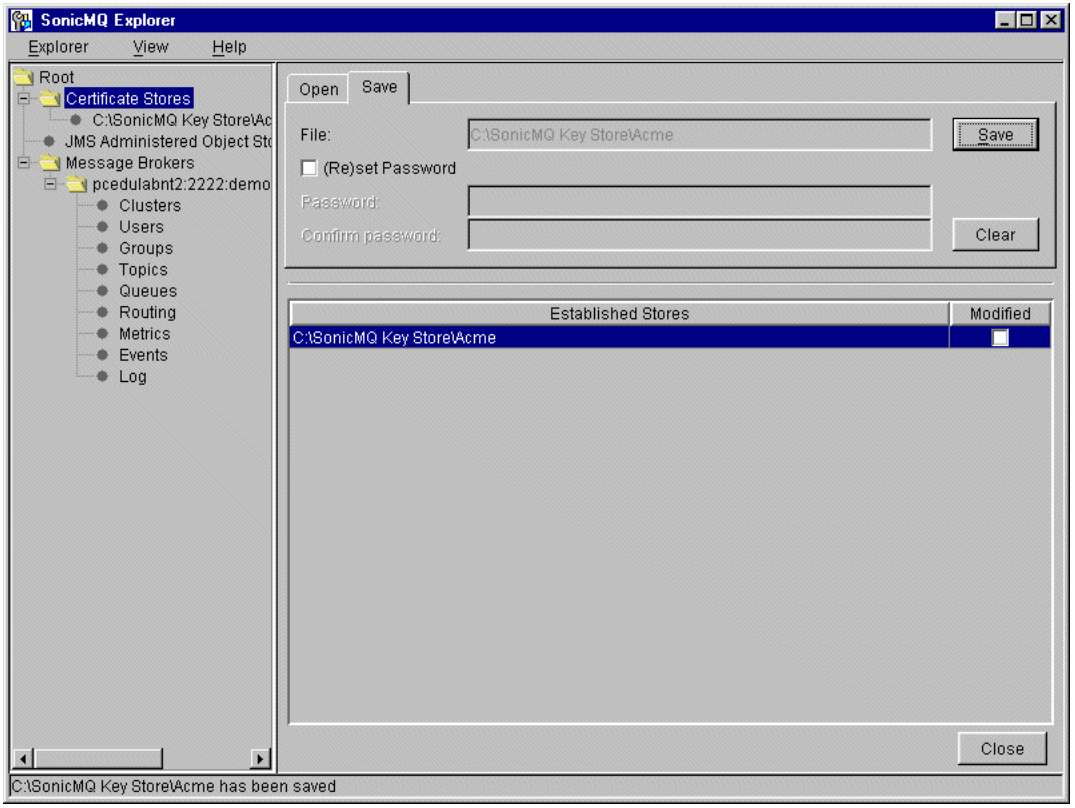


Figure 49. Saving Certificate Stores

If you attempt to exit the Explorer without saving modified stores, it prompts you with a dialog box for each store that needs to be saved, offering you the option to save it.

Using Certificate Stores

After you have opened a Certificate Store, you can perform the following certificate operations for the store:

- Generate a certificate request for review by a Certificate Authority (CA)
- Load approved certificates returned from the CA into the Certificate Store
- Write certificates from the Certificate Store to a file for export
- View the information for each certificate entry in the Certificate Store

➤ To perform a certificate operation:

1. Select the node for an open Certificate Store in the tree view.
2. Select the tab in the right-hand panel associated with the operation you want to perform and complete the specified actions.

Requests Tab

When you select the **Requests** tab, the Explorer allows you to generate a Key Pair and a Certificate Signing Request (CSR), as shown in Figure 50. The right side of the window displays two panels.

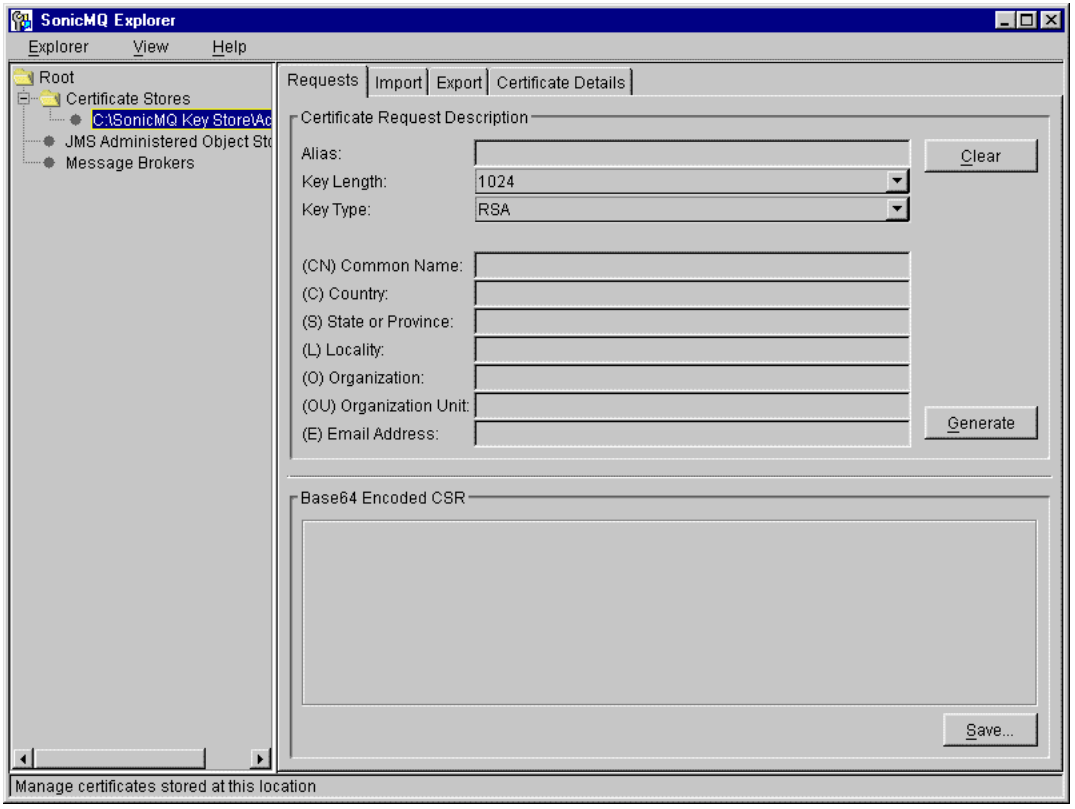


Figure 50. Requests Tab

The top panel, **Certificate Request Description**, contains a number of fields:

- **Alias** — A unique id that you assign, which is used as an index for looking up the data within the Certificate Store
- **Key Length** — Specifies the length of the key, in bits

- **Key Type** — Used to determine the type of key pair that is generated from the following choices:
  - **RSA**
  - **DSA**
- **Relative Distinguished Name (RDN) data** — Several fields used to hold the data for creating the certificate's RDN

The bottom panel, **Base64 Encoded Text**, consists of a text field where the generated CSR is displayed. This CSR can be either cut-and-pasted into a Certificate Authority's tool, or the contents can be saved to a disk file and sent to the Certificate Authority (CA) by whatever means you prefer.

Once you have provided information requested in the **Certificate Request Description** tab, select **Generate** to create a PKCS8-formatted Private/Public Key Pair and generate the corresponding PKCS10-formatted CSR. The information and Private Key are also placed in the Certificate Store using **Alias** as an index.

Note The longer the value you specify for **Key Length**, the longer it takes to generate the key pair. While the tool generates the key pair, it displays **Working ...** in the status bar.

Import Tab

When you select the **Import** tab, the Explorer allows you to load the Certificates returned from the CA, as shown in Figure 51. The **Import** tab can be used to

import a Certificate Chain or an individual Certificate. The right side of the window displays two panels.

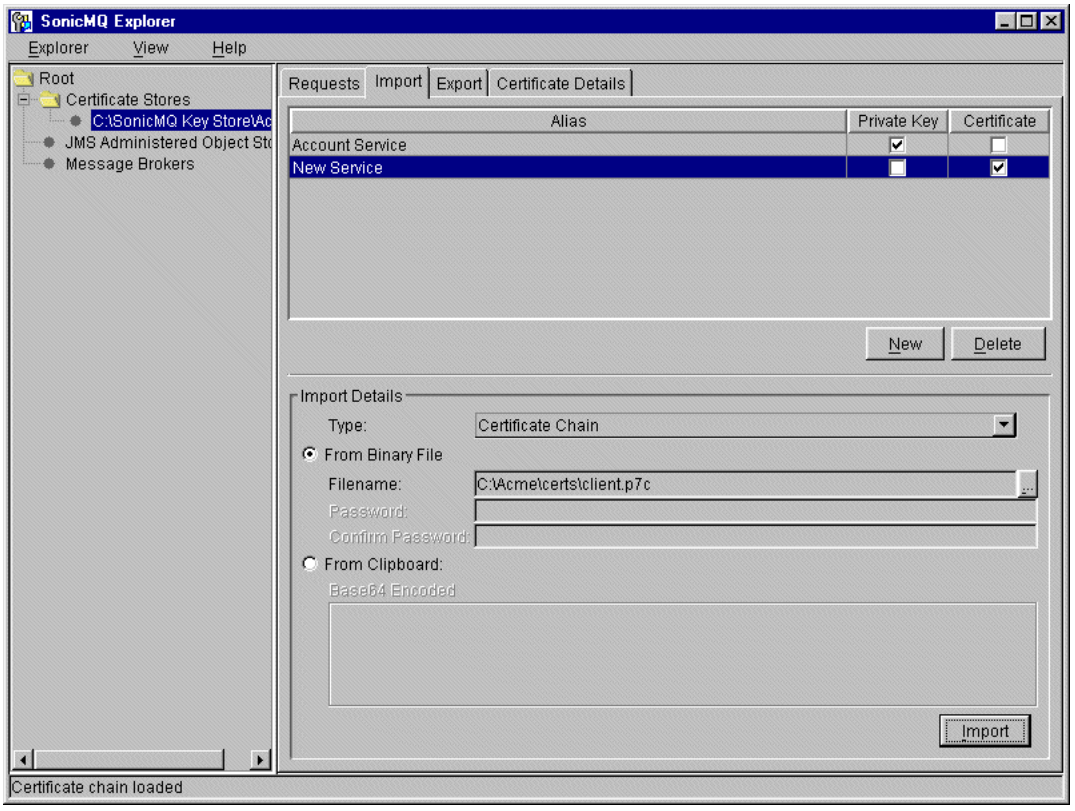


Figure 51. Import Tab

The top panel displays a table for viewing the existing contents of the Certificate Store. The table lists the aliases, a column indicating if the store contains a Private Key for the entry, and a column showing whether the Certificate or Certificate Chain has been loaded for this entry:

- Select **New** to create new entries in the table. If the original CSR had been generated using the Explorer Certificate Management Tool, an entry already exists in the table. Otherwise, you must create a new entry and specify an **Alias**.
- Select **Delete** to delete the entire entry from the Certificate Store.

The bottom panel is used to provide the Import certificate information:

- You must specify the Certificate **Type** to identify whether the data being imported is a Certificate, a Certificate Chain, or a PKCS8 Private Key.
- Use the radio buttons to specify whether the Certificates are imported from a file, or from base64 encoded text cut-and-pasted from the system clipboard. For files, a browse button is provided to allow you to locate the file.
- If you are importing an existing PKCS8 Private Key, you must enter and confirm the password. Note that you can only import private keys from a file.
- Select **Import** to load the certificate into the Certificate Store.

Export Tab

When you select the **Export** tab, the Explorer allows you to export the Certificates, as shown in Figure 52. The right side of the window displays two panels.

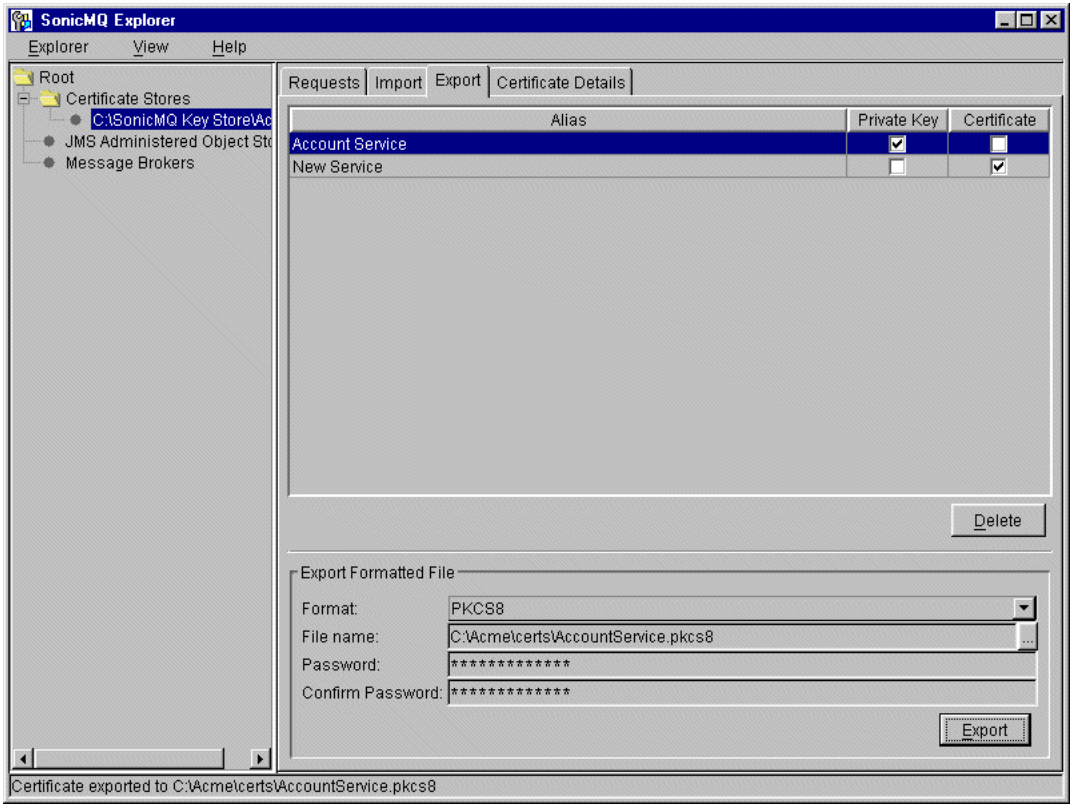


Figure 52. Export Tab

The top panel displays a table for viewing the contents of the Certificate Store. The table lists the aliases, a column indicating if the store contains a Private Key for the entry, and a column showing whether the Certificate has been loaded for this entry.

Select an entry to export from the table in the top panel. The format of the exported file varies depending on the data within the table:

- If the CSR was generated using these tools, then the Private Key is available, and the exported file will be in PKCS7 or PKCS12 format.
- If there is no Private Key in the Certificate Store for this entry and the certificate is a Certificate Chain, then the exported file will be in PKCS7 format.
- If there is no Private Key for this entry and the certificate is a single certificate, then the exported file will be in .DER binary format.

The tool displays the valid choices for the export format based on the alias selected. If there is no certificate or certificate chain, nothing can be exported.

The bottom panel is used to provide the **Export Formatted File** information:

- Specify the **File name** where the file will be exported.
- Specify a **Password** for PKCS12-formatted files, which are password encrypted.
- Select **Export** to write the file to disk.

Note Specify the **File name** and **Password** generated by this tool in `broker.ini` as `SSL_CERTIFICATE_CHAIN` and `SSL_PRIVATE_KEY_PASSWORD`.

- Select **Delete** to delete the entire entry from the Certificate Store.

Certificate Details Tab

When you select the **Certificate Details** tab, the Explorer allows you to view the RDN and other certificate information for an entry in the Certificate Store, as shown in Figure 53.

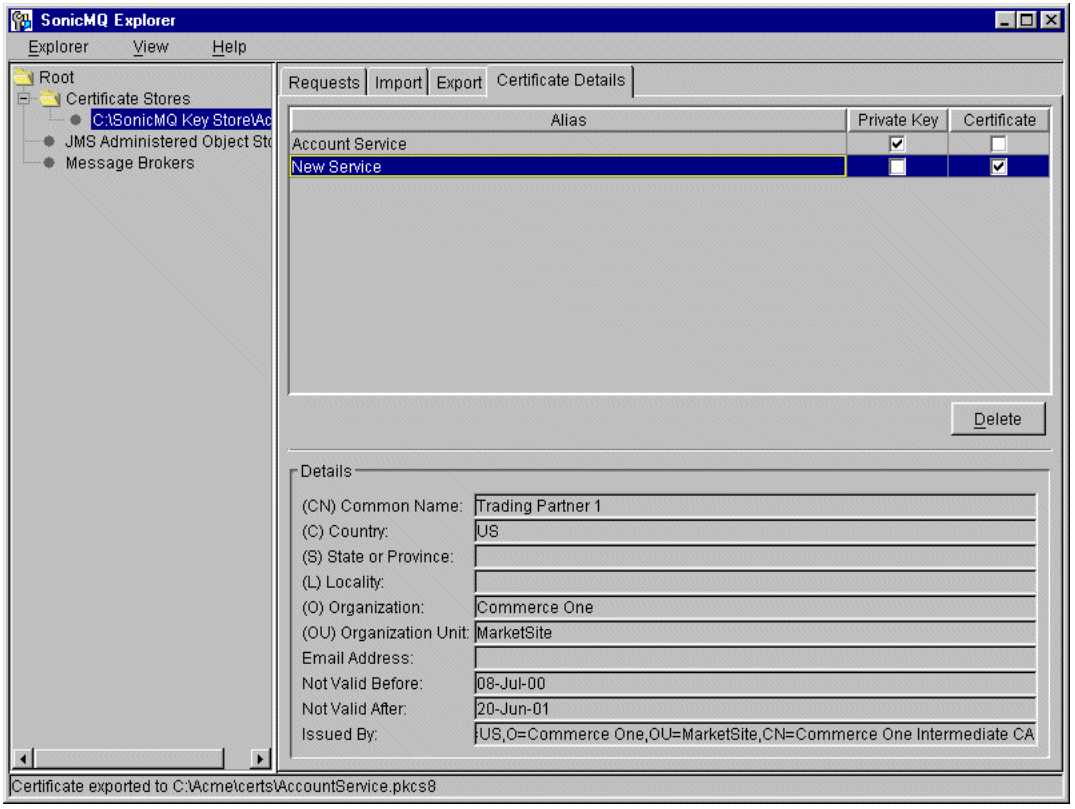


Figure 53. Certificate Details Tab

Individual fields are editable so that you can scroll and view long lines of text, such as the **Issued By** field. However, any changes you make are not saved. To make and save changes, the entire entry must be deleted and a new one created to replace it.

Select an entry from the table in the top panel to populate the **Details** panel.

Chapter 4 Command-line Administration Tool

Admin Tool, also called the Admin shell, or Admin for short, is a command-line tool that provides similar functionality to Explorer. Since it is command-driven, it gives you the option of writing scripts to automate repetitive procedures. In this chapter, Admin is described in the following sections:

- "Starting and Stopping Admin Tool" on page 190.
- "Customizing Admin Tool Behavior" on page 190 describes how to adjust the line length and scrolling behavior for tool output.
- "Command-line Help" on page 192 describes how to obtain online listings of all commands together with brief descriptions.
- "Running Admin with a Script" on page 193 tells you how to run the Admin tool with a script.
- "Administration Commands" on page 194 lists all commands alphabetically and describes each command by function area.

Starting and Stopping Admin Tool

You start Admin, the command-line tool, in different ways, depending on whether you have a Windows, UNIX, or Linux installation.

➤ **To start Admin:**

- ☐ On Windows, choose **Start > Progress SonicMQ > Admin Shell**.
- ☐ On UNIX or Linux, run the `admin.sh` script.

A command window with the command prompt **Admin>** appears.

➤ **To stop Admin:**

- ☐ Enter `bye` at the **Admin>** prompt.

Customizing Admin Tool Behavior

You can modify the display behavior of the Admin tool by specifying:

- The scrolling behavior when showing the server log
- The number of characters per line to print when displaying help

You do this by setting the properties `admin.scroll` and `admin.wrap` in the script that brings up the Admin tool (`admin.bat` for Windows or `admin.sh` for UNIX or Linux).

admin.echo

`admin.echo={true|`**`false`**`}`

This property determines if commands will be echoed to `stdout`. If you are using a script file, you should probably set this to `true` so that the commands will be displayed. When using Admin interactively, you should set this to `false` to avoid showing commands twice.

admin.prompt

`admin.prompt={`**`true`**`|false}`

This property determines if the **Admin>** prompt will be displayed. You should set it to false to keep the prompt from being displayed when Admin input is coming from a script file.

admin.reqTimeout

admin.reqTimeout={**30**|*seconds*}

This property sets the number of seconds that Admin Tool will wait for a response to a request before timing out. If a timeout occurs an error message will be displayed.

admin.scroll

admin.scroll={**22**|*number_of_lines*}

This property specifies how many lines SonicMQ prints before pausing when printing the server log with the command `show broker log`.

admin.wrap

admin.wrap={**79**|*number_of_characters*}

This property specifies how many characters per line to print when displaying command help.

Setting Admin Tool Properties

You can set an Admin property by prefacing it with `-D` and placing it before the name of the class `progress.message.tools.Admin` in the script's Java command.

For example, suppose you wish to change `admin.wrap` to 50.

On Windows, you change the line in `admin.bat` from

```
"%SONICMQ_JRE%" -cp \
"%JRE_CLASSPATH%;%SONICMQ_LIB%\broker.jar;%SONICMQ_LIB%\client.jar" \
progress.message.tools.Admin
```

to

```
"%SONICMQ_JRE%" -cp \
"%JRE_CLASSPATH%;%SONICMQ_LIB%\broker.jar;%SONICMQ_LIB%\client.jar" \
-Dadmin.wrap=50 progress.message.tools.Admin
```

On UNIX or Linux, you change the line in `admin.sh` from

```
$SONICMQ_JRE -classpath "$JRE_CLASSPATH:$SONICMQ_LIB/broker.jar" \
progress.message.tools.Admin
```

to

```
$SONICMQ_JRE -classpath "$JRE_CLASSPATH:$SONICMQ_LIB/broker.jar" \
-Dadmin.wrap=50 progress.message.tools.Admin
```

Command-line Help

Online help is available for Admin's command-line interface. It operates similarly to the familiar FTP command. Top-level help is available by entering a question mark (`?`) at the command prompt:

```
Admin> ?
```

This produces a list of the top-level command tokens, as shown in Figure 54.

```
Admin>?

Commands are:

//              add             bye             connect         del
disconnect      reset           set             show            shutdown
store           unstore
```

Figure 54. Top-level Command Tokens

Following a command by a `?` produces a description of the command, while following an incomplete string of command tokens produces a list of all tokens that can follow the given string. For example:

```
Admin> connect ?
```

produces the list of tokens:

- `broker`
- `store`

indicating that `connect` can be followed by `broker` or `store`.

```
Admin> connect store ?
```

produces the list of tokens:

- `file`
- `jndi`

indicating that `connect store` can be followed by `file` or `jndi`. Finally:

```
Admin> connect store jndi ?
```

produces a description of the command `connect store jndi`, including the syntax for the command, as shown in Figure 55.

```
Admin>connect store jndi ?

Usage: connect store jndi [<property=value>,...]

Create a connection to a JNDI-conformant naming service where JMS Administered
Objects are stored.

Use a comma-delimited list to specify JNDI connection properties that will
override any such properties supplied on the command line using normal JVM
startup arguments.

This tool supports only a single connection to a JMS Administered Object
store. You can disconnect from a currently connected store in order to connect
to another store without leaving the tool.

See also:

  disconnect store
```

Figure 55. Sample Help Output

Running Admin with a Script

You might want to have script commands echoed to the terminal. If you do, edit the `admin.bat` (Windows) or `admin.sh` (UNIX or Linux) file and add the parameter `-Dadmin.echo=true` to the line that brings up the Admin Tool. This line would now begin:

`"%SONICMQ_JRE%" %SONICMQ_SSL_CLIENT% **-Dadmin.echo=true** -cp`

You might also want to add the the path to *`sonic-install-dir`*`/bin` to your system PATH, so you do not have to change to that directory to run the command.

➤ To run admin with a script:

1. Open a command window.
2. If you have not added *`sonic-install-dir`*`/bin` to your system PATH, `cd` to *`sonic-install-dir`*`/bin`.
3. Issue the command:

   `admin < `*`path/scriptname`* (Windows)

   `admin.sh < `*`path/scriptname`* (UNIX or Linux)

where *scriptname* is the name of a text file containing Admin commands, and *path* is the absolute path to the file or the path relative to the SonicMQ installation directory.

Note

If you want the path to be relative to a directory *scriptdir* different from the SonicMQ installation directory, edit the `admin.bat` or `admin.sh` file so that the line beginning `cd` reads
`cd` *scriptdir-path*
where *scriptdir-path* is the full path and name of *scriptdir*.

Administration Commands

This section lists and describes all commands available in the Admin tool.

Table 13 lists all commands in alphabetical order. After each command you will find the functional area to which the command applies and the kind of connections to which it applies. Commands can apply to a server, to a configuration server, or to a JMS administered object store. The table also shows if the command applies to the Publish and Subscribe (Pub/Sub) domain, to the Point-to-Point (PTP) domain, or to both domains. Finally, the page number on which you can find a description of the command is listed.

The description of the commands are arranged by functional area, in the following subsections:

Note Table 13 indicates that user administration commands, group administration commands, and certain queue and topic administration commands apply to a security-enabled server. If, however, the security-enabled server belongs to a cluster, these commands apply solely to the configuration server for the cluster.

Table 13. Administration Commands

Command Name	*Functional Area*	*Applies to*	*Domain*	*Page*
// (null command)	miscellaneous	command scripts	N/A	215
add cluster	servers and clusters	configuration server	Both	199
add clusterbroker	servers and clusters	configuration server	Both	199
add group	groups	security-enabled server	Both	201
add groupuser	groups	security-enabled server	Both	202
add routing user	routings	security-enabled server	PTP	207
add user	users	security-enabled server	Both	201
bye	miscellaneous	Admin tool	N/A	215
connect broker	servers and clusters	server	Both	199
connect store file	administered objects	JMS administered object store	Both	211
connect store jndi	administered objects	JMS administered object store	Both	211
del broker	servers and clusters	configuration server	Both	200
del cluster	servers and clusters	configuration server	Both	200
del clusterbroker	servers and clusters	configuration server	Both	200
del global	routings	server	PTP	207
del group	groups	security-enabled server	Both	202
del groupuser	groups	security-enabled server	Both	202
del queue	destinations	server	PTP	203
del queue acl	destinations	security-enabled server	PTP	203

Table 13. Administration Commands (*continued*)

Command Name	Functional Area	Applies to	Domain	Page
del queue messages	destinations	server	PTP	203
del queue qop	destinations	security-enabled server	PTP	203
del routing	routings	server or configuration server	PTP	207
del routing user	routings	security-enabled server	PTP	207
del topic acl	destinations	security-enabled server	Pub/Sub	203
del topic durables	destinations	server	Pub/Sub	203
del topic qop	destinations	security-enabled server	Pub/Sub	204
del user	users	security-enabled server	Both	201
disconnect broker	servers and clusters	server	Both	200
disconnect store	administered objects	JMS administered object store	Both	212
reset broker log	servers	log-enabled server	Both	209
reset password	users	security-enabled server	Both	201
set queue	destinations	server	PTP	204
set queue acl	destinations	security-enabled server	PTP	205
set queue qop	destinations	security-enabled server	PTP	205
set routing	routings	server or configuration server	PTP	207
set topic qop	destinations	security-enabled server	Pub/Sub	205
set topic acl	destinations	security-enabled server	Pub/Sub	205
show broker connections	servers and clusters	server	Both	200
show broker events start	servers	server	Both	210
show broker events stop	servers	server	Both	210
show broker log	servers	log-enabled server	Both	211
show broker metrics	servers	server	Both	211

Table 13. Administration Commands (*continued*)

Command Name	Functional Area	Applies to	Domain	Page
show brokers	servers and clusters	configuration server	Both	200
show clusters	servers and clusters	configuration server	Both	200
show globals	routings	server	PTP	208
show groups	groups	security-enabled server	Both	202
show groupusers	groups	security-enabled server	Both	202
show keywords	miscellaneous	Admin tool	N/A	215
show queue acls	destinations	security-enabled server	PTP	205
show queues	destinations	server	PTP	206
show routing users	routings	security-enabled server	PTP	209
show routings	routings	server or configuration server	PTP	209
show store	administered objects	JMS administered object store	Both	212
show topic acls	destinations	security-enabled server	Pub/Sub	206
show topic durables	destinations	server	Pub/Sub	206
show topics	destinations	security-enabled server	Pub/Sub	206
show users	users	security-enabled server	Both	201
show version	miscellaneous	Admin tool	N/A	215
shutdown broker	servers and clusters	server	Both	200
store queue	administered objects	JMS administered object store	PTP	212
store queuefactory	administered objects	JMS administered object store	PTP	212
store topic	administered objects	JMS administered object store	Pub/Sub	213
store topicfactory	administered objects	JMS administered object store	Pub/Sub	213
unstore	administered objects	JMS administered object store	Both	215

In the following pages the administration commands are listed by functional area, and alphabetically within each area. Each command is followed by a line showing its syntax and then by a short explanation of its usage.

In this list, alternative values are enclosed in braces ({}) and delimited by the or symbol (|), and configurable values are shown in *italics*. Items enclosed in square brackets ([]) are optional. Default values are listed in **boldface**. No spaces are allowed before or after the equal sign (=).

Note In the online help, the syntax format differs slightly from that given here. There is no use of italics or boldface. Configurable values are enclosed in angle brackets (<>), default values are enclosed in square brackets ([]), and configurable values are sometimes given different names.

Important Do not use the characters period (.), asterisk (*), pound sign (#), backslash (\), or dollar sign ($) in user, group, server, or cluster names. Do not use wildcard characters (* or #), backslash (\), or double colon (::) in queue or topic names. Do not use dollar sign ($) as the first character of a queue or topic name. A queue or topic name cannot begin with the string "SonicMQ.".

SonicMQ supports all Unicode characters, but characters must be legal for table names in your database. See "SonicMQ and Database Character Sets" on page 24.

If a configurable value is specified and the configurable value contains any spaces, the configurable value must be enclosed in double quotes (" ").

Server and Cluster Administration Commands

The following are all server commands that deal with clusters, connecting to and disconnecting servers, and shutting down servers. Additional server commands are covered in "Miscellaneous Server Commands" on page 209.

add cluster

```
add cluster cluster_name
```

Adds the named cluster *cluster_name* to the list of clusters managed by the configuration server.

Important The *cluster_name* can contain a maximum of 64 Unicode characters. Do not use the characters period (.), asterisk (*), pound sign (#), backslash(\), or dollar sign ($) in *cluster_name*.

Before adding a server to a security-enabled server cluster, you must perform several preliminary steps. See "Adding Servers to Security-enabled Clusters" on page 81 for details.

add clusterbroker

```
add broker cluster_name server_name
```

Adds the existing server *server_name* to the existing cluster *cluster_name*.

connect broker

```
connect broker {tcp://|http://|ssl://}host{:2506|:port}
[user_name [password]]
```

Connects Admin to the server at the location specified by {**tcp://**|http://|ssl://}*host*{**:2506**|:*port*}, where *host* is a resolvable IP name or address and *port* is the port number where the server is listening for incoming connections.

You must provide the *user_name* and *password* of a member of an administrators group (or equivalent) if you wish to perform security functions on a security-enabled server. In this case, you can supply the built-in user name `Administrator`. The default password for `Administrator` is also `Administrator`.

The Admin tool supports only a single connection to a server. You can disconnect from a currently connected server in order to connect to another server without leaving Admin tool.

del broker

```
del broker server_name
```

Deletes server *server_name* from the list of servers using the currently connected server as their configuration server.

del cluster

```
del cluster cluster_name
```

Deletes the named cluster *cluster_name* from the list of clusters managed by the configuration server.

del clusterbroker

```
del clusterbroker cluster_name server_name
```

Removes the server *server_name* from the cluster *cluster_name*.

disconnect broker

```
disconnect broker
```

Disconnects from the currently connected server.

show broker connections

```
show broker connections
```

Displays the current list of connected users and connect IDs for the server.

show brokers

```
show brokers
```

Displays the list of servers using the currently connected server as their configuration server.

show clusters

```
show clusters
```

Displays the list of clusters (and their member servers) managed by the currently connected configuration server.

shutdown broker

```
shutdown broker
```

Shuts down the currently connected server.

Once the server has shut down, the Admin tool will behave as though the command `disconnect broker` had been executed.

User Administration Commands

add user

```
add user user_name [password]
```

Adds the named user *user_name* to the list of users managed by the security-enabled server. The user will be assigned the specified password *password*, if you supply it. Otherwise, the user will be assigned a blank password.

The *user_name* can have a maximum of 64 Unicode characters. Do not use the characters period (.), asterisk (*), pound sign (#), backslash(\), or dollar sign ($) in a username. Do not use the reserved name **AUTHENTICATED**.

del user

```
del user user_name
```

Deletes the user *user_name* from the list of users managed by the security-enabled server.

reset password

```
reset password user_name [password]
```

Resets a user's password to the value specified by *password*. The default is no password. The user *user_name* must exist in the list of users managed by the security-enabled server.

show users

```
show users [prefix]
```

Displays the list of all users managed by the security-enabled server. If you specify a *prefix*, only those users with names that begin with this prefix will be shown.

If the server is not security-enabled, displays the list of users that have durable subscriptions.

Group Administration Commands

add group

```
add group group_name
```

Adds the named group *group_name* to the list of groups managed by the security-enabled server.

The *group_name* can have a maximum of 64 Unicode characters. Do not use the characters period (.), asterisk (*), pound sign (#), backslash(\), or dollar sign ($) in a group name.

add groupuser

```
add groupuser group_name user_name
```

Adds the existing user *user_name* to the existing group *group_name* managed by the security-enabled server.

del group

```
del group group_name
```

Deletes the named group *group_name* from the list of groups managed by the security-enabled server.

del groupuser

```
del groupuser group_name user_name
```

Deletes the user *user_name* from the group *group_name* managed by the security-enabled server.

show groups

```
show groups [prefix]
```

Displays the list of groups managed by the security-enabled server.

If you specify a *prefix*, only those groups with names that begin with this prefix will be shown.

show groupusers

```
show groupusers group_name [prefix]
```

Displays the list of users, including routing users, that are members of the specified group *group_name* being managed by the security-enabled server.

If you specify a *prefix*, only those users with names that begin with this prefix will be shown.

Topic and Queue Administration Commands

The following commands concern the administration of topics and queues.

del queue

`del queue` *queue_name*

Deletes the queue `queue_name`. This command does not delete any associated QoP or ACL entries when the server has security enabled.

del queue acl

`del queue acl` *queue_name* {*group_name*|*user_name*}

Deletes the ACL entry on the specified queue *queue_name* for either the specified group *group_name* or user *user_name*. Queue ACL entries are managed by the security-enabled server.

del queue messages

`del queue messages` *queue_name*

Deletes all the messages currently held in the named queue *queue_name*.

del queue qop

`del queue qop` *queue_name*

Deletes the QoP associated with the specified existing queue *queue_name* managed by the security-enabled server.

del topic acl

`del topic acl` *topic_name* {*group_name*|*user_name*}

Deletes the ACL entry on the specified topic *topic_name* for either the specified group *group_name* or user *user_name*. Topic ACL entries are managed by the security-enabled server.

del topic durables

`del topic durables` *user_name client_id* {**all**|*subname*[,*subname*,...]}

Deletes durable subscriptions for the specified user *user_name* and client identifier *client_id* held by the server.

The default is `all`, indicating all subscriptions for the *user_name-client_id* pair will be deleted. Alternatively, a comma-delimited list of subscribed topics *subname* can be provided for finer granularity.

del topic qop

```
del topic qop topic_name
```

Deletes the QoP associated with the specified existing topic *topic_name* managed by the security-enabled server.

set queue

```
set queue queue_name {local|global} {shared|exclusive}
{1200,1400,1000|retrieve_threshold,save_threshold,max_size}
```

Sets the queue *queue_name* with the supplied attributes. Do not use wildcard characters (* and #), backslash (\), or double colons (::) in *queue_name*. Do not use a dollar sign ($) as the first character of *queue_name*. A queue name cannot begin with the string "SonicMQ.". A queue name can have a maximum of 256 characters. If the queue does not exist the queue will be created.

When queue routing is used, a `global` queue allows other routing servers to write to it. By default, a queue is `local`. A local queue will not allow other routing servers to write to it.

A queue operating mode of `shared` (the default value) allows multiple concurrent receivers. Access to the queue by multiple concurrent receivers is disallowed if the queue operating mode is `exclusive`.

The *retrieve_threshold* is the minimum number of kilobytes of message data that is in memory before messages are retrieved from the database to the specified queue *queue_name*. It defaults to 1200. The *retrieve_threshold* must be less than or equal to the *save_threshold*.

The *save_threshold* is the maximum number of kilobytes of message data enqueued before messages from the queue *queue_name* are saved to the database in order to reduce memory usage. It defaults to 1400.

The *max_size* is the upper limit for the size of a queue in kilobytes. When the queue data grows to this size, publishing to the queue will be delayed until some of the enqueued messages are dequeued or expire, or until the queue is cleared administratively. It defaults to 1000. If either threshold or maximum queue size is specified, the other two values must be specified also.

A queue name that begins with dollar sign ($) refers to a system queue, which is created by SonicMQ. You cannot change the `local`/`global` flag for a system queue.

You cannot change the `shared`/`exclusive` flag for a routing queue.

set queue acl

```
set queue acl queue_name {group_name|user_name} [{+|-}snd] [{+|-}rcv]
```

Sets an ACL entry for either a group *group_name* or user *user_name* on the specified queue *queue_name*. If the ACL entry does not exist, the entry will be created.

The ACL entry can allow positive (+) or negative (-), send (snd) or receive (rcv) access to the queue. The default—for both send and receive—is inherited access.

set queue qop

```
set queue qop queue_name {none|integrity|privacy}
```

Sets the queue *queue_name* with the supplied QoP. The queue QoP can be specified as none (the default value), integrity, or privacy. If the server does not have security enabled, the QoP setting will be ignored.

set topic qop

```
set topic topic_name {none|integrity|privacy}
```

Sets the specified *topic_name* with the supplied Quality of Protection attribute.

The topic's QoP value can be specified as none (the default value), integrity or privacy. If the server does not have security enabled, the setting will be ignored.

set topic acl

```
set topic acl topic_name {group_name|user_name} [{+|-}pub] [{+|-}sub]
```

Sets an ACL entry for a group *group_name* or user *user_name* on the specified topic *topic_name*. If the ACL entry does not exist, the entry will be created.

The ACL entry can grant (+) or deny (-), publish (pub) or subscribe (sub) access to the topic. The default—for both send and receive—is inherited access.

show queue acls

```
show queue acls queue_name
```

Displays a list of ACL entries for the specified queue *queue_name* managed by the security-enabled server. For each ACL entry, the group/user and positive/negative/inherit send and receive rights will be displayed.

show queues

`show queues [prefix] [all]`

Displays a list of all the static queues managed by the server. If you specify a *prefix*, only those queues with names that begin with this prefix will be shown. To include system queues for display, use the keyword `all`.

Displays the queue visibility (local/global), the access mode (exclusive/shared), the retrieve and save extents (bytes), the max queue size (bytes), and the number of messages currently in the queue for each queue.

If the server has security enabled, an additional list of QoP settings will also be displayed.

show topic acls

`show topic acls topic_name`

Displays a list of ACL entries for the specified topic *topic_name* managed by the security-enabled server. For each ACL entry, the group/user and positive/negative/inherit publish and subscribe rights will be displayed.

show topic durables

`show topic durables user_name`

Displays the list of durable subscriptions for the specified user *user_name* managed by the server. Each list entry will display the client identifier, subscription name, and topic name.

show topics

`show topics [prefix] [all]`

Displays the list of topics managed by the security-enabled server.

If you specify a *prefix*, only those topics with names that begin with this prefix will be shown. For each topic the associated QoP will be displayed. To include system topics for display, use the keyword `all` .

Routing Administration Commands

The following commands apply to Dynamic Routing Architecture (DRA).

add routing user

```
add routing user user_name routing_node [password]
```

Adds the special named user *user_name* (used for authenticating routing connections between servers) to the list of routing users managed by the security-enabled server. The *routing_node* indicates the routing node name from which the inbound connection will originate.

The routing user will be assigned the specified *password* if you supply it, otherwise the user will be assigned a blank password. The password will not be checked over an SSL connection when *user_name* is the certificate identity.

del global

```
del global routing_node::global_queue
```

Deletes the named global destination *routing_node*::*global_queue* from the routing system.

del routing

```
del routing routing_node
```

Deletes the named *routing_node* routing connection.

del routing user

```
del routing user user_name
```

Deletes the existing routing user *user_name* from the list of users managed by the server.

set routing

```
set routing routing_node "connection_URL[,connection_URL,...]"
user_name {password|keep} {nolb|lb} {sequential|random} {timeout|300}
{noadvertise|advertise} {dynamic|static} [server_name]
```

Sets a routing connection in a server. If the routing connection exists, it will be modified; if it does not exist, it will be created. Routing connections are defined at stand-alone servers or cluster configuration servers. The name *routing_node* identifies the routing destination node to which the connection will be made.

You must provide at least one routing connection path (message server connection URL) *connection_URL*. If you need multiple possible connection paths, specify them as a comma-delimited list enclosed in double-quotes (").

You must provide a *user_name* and *password* for the connections. Use "" to indicate an empty value or no value. To update an existing routing connection without resetting the password, specify keep to retain the existing password.

When a connection is to a routing node that is a cluster of servers, it will be load-balanced if you specify the parameter lb. The default nolb is for no load-balancing.

If you specify sequential, the list of server connection URLs will be attempted sequentially. If you omit the parameter or specify random, the choice will be random from the list.

The *timeout* value defines the length of time in seconds before an inactive routing connection is disconnected. A value of 0 (zero) represents an infinite timeout.

If you specify advertise, information about all known global queues in the node will be sent to the routing destination node for dynamic routing configuration purposes. If you specify noadvertise (the default value), this information is not sent.

If you specify static, the routing connection will always be used to route messages to the routing destination node. If you specify dynamic (the default value), connections initiated from the routing destination node can be used for routing subsequent messages. For static routing connections the *server_name* determines the server from which the routing connection will be originated. For a nonclustered server, this defaults to the local server name.

show globals

show globals [*prefix*]

Displays a list of global destinations known by the connected server in *routing_node*::*global_queue* format. This information is used by the server to route messages to their destinations and is retained across server shutdowns.

If you specify a *prefix*, only those global destinations with names that begin with this prefix will be shown.

show routing users

```
show routing users [prefix]
```

Displays the list of routing users managed by the server with the associated routing node name for each routing user. If you specify a *prefix*, only those routing users with names that begin with this prefix will be shown.

show routings

```
show routings [prefix]
```

Displays the list of routing connections managed by the server or configuration server. If you specify a *prefix*, only those routing connections with names that begin with this prefix will be shown.

For each routing connection, the following information is displayed:

- Routing node name
- Connection URLs
- Connection user
- Connect load-balance flag (`lb` or `nolb`, that is load-balancing or no load-balancing)
- Connection attempt order flag (`sequential` or `random`)
- Connection timeout, in seconds
- Global queue advertising flag (`advertise` or `noadvertise`)
- Static routing flag (`static` or `dynamic`)
- Routing server name, if defined

Miscellaneous Server Commands

The following commands are those dealing with servers that were not covered in "Server and Cluster Administration Commands" on page 198.

reset broker log

```
reset broker log
```

Truncates the error log of the currently connected server.

show broker events start

```
show broker events start {all|event_id[,event_id,...]}
```

Starts asynchronously displaying administration events sent by the server. Either `all` events or a comma-delimited list of administration events *`event_id,...`* should be specified. Events are displayed to the Admin tool console.

The following administration event identifiers are supported:

- **`connect`** — Client connection established
- **`disconnect`** — Client disconnection occurred
- **`reject`** — Client login request rejected
- **`drop`** — Client connection aborted
- **`redirect`** — Client connection redirected
- **`undelivered`** — Message transferred to Dead Message Queue (DMQ)
- **`dmqstatus`** — Message added to the DMQ, and the DMQ has exceeded a user-specified fraction of its maximum size

show broker events stop

```
show broker events stop {all|event_id[,event_id,...]}
```

Stops displaying administration events sent by the server. Either `all` events or a comma-delimited list of administration events *`event_id, . . .`* should be specified.

The following administration event identifiers are supported:

- **`connect`** — Client connection established
- **`disconnect`** — Client disconnection occurred
- **`reject`** — Client login request rejected
- **`drop`** — Client connection aborted
- **`redirect`** — Client connection redirected
- **`undelivered`** — Message transferred to Dead Message Queue (DMQ)
- **`dmqstatus`** — Message added to the DMQ, and the DMQ has exceeded a user-specified fraction of its maximum size

show broker log

```
show broker log
```

Displays the server's current error log.

show broker metrics

```
show broker metrics
```

Displays a list of name-value pairs for the metrics captured by the server.

JMS Administered Object Store Administration Commands

connect store file

```
connect store file pathname
```

Creates a pseudo-connection to a directory where JMS administered objects are stored. The directory path *pathname* must exist and be reachable through the operating system.

The Admin tool supports only a single connection to a JMS administered object store. You can disconnect from a currently connected store in order to connect to another store without leaving the tool.

connect store jndi

```
connect store jndi [property=value,...]
```

Creates a connection to a JNDI-conformant naming/directory service where JMS administered objects are stored.

Use a comma-delimited list to specify JNDI connection properties that will override any such properties supplied on the command line using standard JVM startup arguments. At a minimum, the initial context must be specified.

For example, you might enter the following command (as a single line, ***with no line breaks***):

```
connect store jndi java.naming.provider.url="ldap://LDAP_svr1:389/
ou=TopicConnectionFactories,ou=Message Service,ou=SonicMQ,
dc=progress,dc=com",java.naming.factory.initial=
com.jndi.ldap.LdapCtxFactory
```

The Admin tool supports only a single connection to a JMS administered object store. You can disconnect from a currently connected store in order to connect to another store without leaving the tool.

disconnect store

```
disconnect store
```

Disconnects from the currently connected JMS administered object store.

show store

```
show store
```

Displays a list of JMS administered objects stored at the currently connected store location.

The name under which the object was stored and the JMS administered object type (Topic, Queue, TopicConnectionFactory, or QueueConnectionFactory) are always displayed. Other information displayed will be particular to the JMS administered object type.

store queue

```
store queue object_name queue_name
```

Stores a JMS `javax.jms.Queue` object in the currently connected JMS administered object store. The queue object will be stored under the name *`object_name`* with the specified queue name *`queue_name`*.

If the name already exists in the store, the existing object will be overwritten.

store queuefactory

```
store queuefactory object_name "connection_URL_list" user_name
{password|keep} {connect_id|null} {client_id|null} {sequential|random}
{lb|nolb}
```

Stores a JMS `javax.jms.QueueConnectionFactory` object in the currently connected JMS administered object store. The QueueConnectionFactory object will be stored under the specified name *`object_name`* with the specified properties as follows:

- ***`connection_URL_list`*** — A comma-delimited list of one or more connection URLs. If the list contains just one connection URL, the surrounding double quotes (**"**) are optional. Each connection URL is of the form {**tcp://**|ssl://|http://}*host*{**:2506**|:*port*} where:
  - *host* is a resolvable IP name or address.
  - *port* is the port number for the connection.

  - tcp, ssl, and http are possible connection protocols. SSL is only supported for SonicMQ Professional Developer and E-Business Editions.

- ***user_name*** — The default user.
- ***password*** — The default user's password. Specify keep to retain the existing password.
- ***connect_id*** — The connection identifier. By setting *connect_id* to null you let a single client establish multiple simultaneous connections to the server. By setting connect_id to any other value, you let the specified client establish only a single simultaneous connection.
- ***client_id*** — The JMS client identifier. The value null means the user can set ClientID at connection time, through a setClientID method on the connection. If the factory has a ClientID other than null, it cannot be changed by the client using the ensuing connection.
- **sequential** or **random** — If sequential (the default), the first URL in *connection_URL_list* will be tried first. If random, random selection determines which URL will be tried first. In either case, subsequent connection attempts will be made from the list in sequential order.
- **lb** or **nolb** — If lb, the requested connection can be redirected for load balancing. If nolb (the default), the requested connection cannot be redirected.

If the name already exists in the store, the existing object will be overwritten.

store topic

store topic *object_name topic_name*

Stores a JMS javax.jms.Topic object in the currently connected JMS administered object store. The topic object will be stored under the name *object_name* with the specified topic name *topic_name*.

If the name already exists in the store, the existing object will be overwritten.

store topicfactory

store topicfactory *object_name* "*connection_URL_list*" *user_name password* {*connect_id*|null} {*client_id*|null} {**sequential**|random} {lb|**nolb**}

Stores a JMS javax.jms.TopicConnectionFactory object in the currently connected JMS administered object store. The TopicConnectionFactory object

will be stored under the name *object_name* with the specified properties as follows:

- ***connection_URL_list*** — A comma-delimited list of one or more connection URLs. If the list contains just one connection URL, the surrounding double quotes (") are optional. Each connection URL is of the form {**tcp://**|ssl://|http://}*host*{**:2506**|:*port*} where:
  - *host* is a resolvable IP name or address.
  - *port* is the port number for the connection.
  - tcp, ssl, and http are possible connection protocols. SSL is only supported for SonicMQ Professional Developer and E-Business Editions.
- ***user_name*** — The default user.
- ***password*** — The default user's password. Specify keep to retain the existing password.
- ***connect_id*** — The connection identifier. By setting *connect_id* to null you let a single client establish multiple simultaneous connections to the server. By setting connect_id to any other value, you let the specified client establish only a single simultaneous connection.
- ***client_id*** — The JMS client identifier. The value null means the user can set ClientID at connection time, through a setClientID method on the connection. If the factory has a ClientID other than null, it cannot be changed by the client using the ensuing connection.
- **sequential** or **random** — If sequential (the default), the first URL in *connection_URL_list* will be tried first. If random, random selection determines which URL will be tried first. In either case, subsequent connection attempts will be made from the list in sequential order.
- **lb** or **nolb** — If lb, the requested connection can be redirected for load balancing. If nolb (the default), the requested connection cannot be redirected.

If the name already exists in the store, the existing object will be overwritten.

unstore

```
unstore object_name
```

Removes the specified name *object_name* and its associated JMS administered object from the currently connected store.

General Commands

The following commands deal with the Admin Tool itself.

// (double forward slashes)

```
// string
```

Signifies a null command when placed at the start of the command. Commands starting with // are ignored (except that the command // ? produces a description of the command) and are intended for comments in command scripts.

bye

```
bye
```

Ends the current administration session and returns to the operating system shell.

show keywords

```
show keywords
```

Displays the list of keywords that Admin recognizes.

show version

```
show version
```

Displays version information for the Admin tool.

Index

Symbols

A

B

C

D

E

F

G

I

J

L

M

N

O

P

Q

R

S

T

U

V

W