Tivoli Policy Director for MQSeries

Administration Reference Guide

Version 3.8.1

Tivoli Policy Director for MQSeries

Administration Reference Guide

Version 3.8.1

Note
Before using this information and the product it supports, read the information in Appendix E, "Notices" on page 161.

Second Edition (April 2002)

This edition replaces GC32-0809-00

Contents

Preface

Welcome to the *Tivoli® Policy Director for MQSeries® Administration Reference Guide, Version 3.8.1*. Tivoli Policy Director for MQSeries is an extension of Tivoli SecureWay® Policy Director, and it provides a security solution for IBM® MQSeries messages. Tivoli Policy Director for MQSeries allows MQSeries applications to send data with privacy and integrity by using keys associated with the sending and receiving applications. The Tivoli Policy Director master policy server provides centrally defined access control information that Tivoli Policy Director for MQSeries uses to determine which users can and cannot get access to messages in the queues. Tivoli Policy Director for MQSeries performs the following functions:

- Defines and enforces centralized authorization policies, including data protection, for MQSeries® resources (queues and messages in those queues) using the Tivoli Policy Director infrastructure. This infrastructure provides:
  - A common, scalable, and reliable policy repository
  - An extendable resource name space and extendable permission sets
  - A common console for managing policy
- Protects MQSeries data as it flows across the network and as it sits in the queue, using public key infrastructure technology.
- Provides transparent security for existing MQSeries applications. MQSeries applications do not need to change for Tivoli Policy Director for MQSeries to protect them.

IBM MQSeries provides the following items:

- Simple, multiplatform application programming interface (API)
- Assured message delivery
- Time-independent processing
- Partner applications that have an independent state
- Application parallelism

You can use Tivoli Policy Director for MQSeries protection in conjunction with MQSeries built-in security (for example, the Object Authority Manager and the Message Channel exits).

Who Should Read This Guide

The target audience for this guide is system administrators who are familiar with MQSeries on distributed platforms.

Prerequisite Documents

Tivoli Policy Director for MQSeries Release Notes, Version 3.8.1, provides information about the following topics:

- System requirements
- Installation notes
- Defects, limitations, and workarounds
- Documentation additions
- Documentation corrections

The following documents provide information specific to Tivoli Policy Director and its component, Web Portal Manager:

- *Tivoli SecureWay Policy Director Base Installation Guide*
- *Tivoli SecureWay Policy Director Base Administration Guide*
- *Tivoli SecureWay Policy Director Web Portal Manager Administration Guide*

Related Documents

The following are related documents:

- *Tivoli Policy Director Base Developer Reference*
- *IBM MQSeries Planning Guide*
- *IBM MQSeries System Administration Manual*
- *MQSeries Queue Manager Clusters*

What This Guide Contains

The *Tivoli Policy Director for MQSeries Administration Reference Guide, Version 3.8.1* contains the following sections:

- Chapter 1, "Understanding Tivoli Policy Director for MQSeries" on page 1

  Lists Tivoli Policy Director for MQSeries functions and describes key components and dependencies.
- Chapter 2, "Installation and Uninstallation of Tivoli Policy Director for MQSeries" on page 7

  Describes the installation and uninstallation of Tivoli Policy Director for MQSeries.
- Chapter 3, "Configuration" on page 15

  Describes the configuration of Tivoli Policy Director for MQSeries.
- Chapter 4, "Migration to Tivoli Policy Director for MQSeries, Version 3.8.1" on page 29

  Describes the migration of a previous version of Tivoli Policy Director for MQSeries to the current version.
- Chapter 5, "Administering Tivoli Policy Director for MQSeries" on page 33

  Describes the details of deploying Tivoli Policy Director for MQSeries in a typical MQSeries environment.
- Chapter 6, "Managing Identities for Tivoli Policy Director for MQSeries" on page 43

  Discusses assignment of operating system users to Public Key Infrastructure identities to Tivoli Policy Director for MQSeries users.
- Chapter 7, "MQSeries Considerations" on page 63

  Discusses interoperational considerations for MQSeries and Tivoli Policy Director for MQSeries.
- Chapter 8, "Tivoli Policy Director for MQSeries Error Handling" on page 67

  Describes how the Tivoli Policy Director for MQSeries error queue works.
- Chapter 9, "Auditing Tivoli Policy Director for MQSeries" on page 69

  Describes the Tivoli Policy Director audit function support for Tivoli Policy Director for MQSeries.
- Chapter 10, "Serviceability, Debugging and Tracing Messages" on page 77

  Describes and explains the syntax of the serviceability, debugging, and tracing messages.

- Appendix A, “Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager” on page 81

  Describes how to administer Tivoli Policy Director for MQSeries using the Tivoli Policy Director Web Portal Manager.
- Appendix B, “Problem Determination” on page 121

  Lists all the Tivoli Policy Director for MQSeries error messages and detailed information about each message.
- Appendix C, “Quick Start” on page 151

  Describes how to use the sample files to quickly configure Tivoli Policy Director for MQSeries and to test your installation.
- Appendix D, “Hardware Accelerator Support” on page 159

  Describes the support provided for the hardware accelerator cards.
- Appendix E, “Notices” on page 161

  Provides licensing, copyright, and trademark information.

Conventions Used in This Guide

This guide uses several typeface conventions for special terms and actions. These conventions have the following meaning:

Bold	Commands, keywords, authorization roles, and other information that you must enter exactly as shown appear in this guide in **bold** type. Also, the names of other controls appear in **bold** type.
Italics	Variables and values that you must provide and words and phrases that are emphasized appear in *italics*.
`Monospace`	Code examples, output, file names, and system messages appear in `monospace` font.
`[ ]`	Identifies optional arguments. Arguments not enclosed in brackets are required.
`|`	Indicates mutually exclusive information. You can use the argument to the left of the separator or the argument to the right of the separator. You cannot use both arguments in a single use of the command.

This guide uses the UNIX® convention for specifying environment variables and for directory notation. When you use the Microsoft® Windows NT® command line, replace **$***variable* with **%***variable***%** for environment variables and replace each forward slash (**/**) with a backslash (\) in directory paths.

Note: When you use the bash shell on a Windows NT system, you can use the UNIX conventions.

Platform-Specific Information

The following table identifies the supported platform versions known at the time of publication. For more detailed and up-to-date information, see *Tivoli® Policy Director for MQSeries Release Notes, Version 3.8.1*.

Platform	Supported Versions
Solaris	Sun SPARC series running Solaris version 7 or 8

Platform	Supported Versions
Windows NT	IBM-compatible PCs 486 or higher running with Microsoft Windows NT, Version 4.0, Service Packs 6 and later
Windows® 2000	IBM-compatible PCs 486 or higher running with Microsoft® Windows 2000, Service Pack 2
AIX®	IBM RS/6000® series running AIX Version 4.3.3

Accessing Publications Online

You can access all the information in this section online without a user ID or password.

You can access the *Tivoli Customer Support Handbook* at the following Web site:

http://www.tivoli.com/support/handbook/index.html

You can find other Tivoli publications at the following Web site:

http://www.tivoli.com/support/documents

This Web site offers answers to frequently asked questions (FAQs). It also has the Tivoli Information Center page, where you can find product manuals, and the Other Technical Information section, which includes redbooks and white papers. The documentation for some products is available in both PDF and HTML formats. Translated documents are also available for some products.

The following Web site has information for resellers about obtaining Tivoli technical documentation and support:

http://www.tivoli.com/support/smb/index.html

For information for Business Partners about obtaining Tivoli technical documentation, refer to "Ordering Publications".

Ordering Publications

You can order many Tivoli publications online at the following Web site:

http://www.elink.ibmlink.ibm.com/public/applications/ publications/ cgibin/pbi.cgi

You can also order by telephone by calling one of these numbers:

- In the United States: 800 879-2755
- In Canada: 800-426-4968

Customers outside the U.S. can refer to the following Web site to find customer support telephone numbers:

http://www.tivoli.com/support/locations.html

Providing Feedback about Publications

We are very interested in hearing about your experience with Tivoli software and documentation, and we welcome your suggestions for improvements. If you have comments or suggestions about our software and documentation, contact us in one of the following ways:

- Send an e-mail to **pubs@tivoli.com**.
- Fill out our customer feedback survey at the following Web site:

  http://www.tivoli.com/support/survey

Contacting Customer Support

If you have a problem with any Tivoli product, you can contact Tivoli Customer Support. See the *Tivoli Customer Support Handbook* at the following Web site:

http://www.tivoli.com/support/handbook/index.html

The handbook provides information about how to contact Tivoli Customer Support, depending upon the severity of your problem, and the following information:

- Registration and eligibility
- Telephone numbers and e-mail addresses, depending on the country in which you are located
- What information you should gather before contacting support

Chapter 1. Understanding Tivoli Policy Director for MQSeries

Tivoli Policy Director for MQSeries operates in conjunction with Tivoli SecureWay Policy Director to provide a security solution for the IBM MQSeries product. With Tivoli Policy Director for MQSeries you can:

- Secure sensitive or high-value messages processed by IBM MQSeries.
- Control which users have access to specific queues.
- Detect and remove rogue or unauthorized messages before they are processed by a receiving application.
- Generate detailed audit records showing which messages were expressly authorized and encrypted.
- Define authorization and data protection policies centrally for MQSeries resources (queues and messages on those queues) using a Web browser or command line.
- Provide integrity and privacy protection for your data as it flows across the network and while it is in a queue.
- Secure existing off-the-shelf and customer-written applications for IBM MQSeries.

Tivoli Policy Director for MQSeries furnishes MQSeries applications with the following functionality:

- A centralized authorization service that defines access control policies for MQSeries queues and messages in these queues.
- Privacy, in the form of encryption, and integrity, in the form of checks against message modification, so that senders and receivers of MQSeries messages can exchange themwith security. Tivoli Policy Director for MQSeries provides these services while the messages are in transit as well as when the messages are stored in the queues.
- Integration of public key infrastructure (PKI) technology into MQSeries. Tivoli Policy Director for MQSeries identifies MQSeries users with identities that are operating system and network independent.
- Transparent message-level security. MQSeries applications do not have to be modified to be protected by Tivoli Policy Director for MQSeries.

This chapter contains the following sections:

Tivoli Policy Director for MQSeries Compatibility

Although Tivoli Policy Director for MQSeries, Version 3.8.1, depends on several technology components to provide a security infrastructure, it does not require you to license any additional Tivoli software to use this solution. Tivoli Policy Director for MQSeries supports the following certificate authorities:

- Tivoli Public Key Infrastructure, Version 3.7.1
- Entrust WebConnector, Version 5.0
- iPlanet Certificate Management System (CMS), Version 4.2
- PKIServ application for Resource Access Control Facility (RACF®) on OS/390® Release 10
- Baltimore UniCERT, Version 3.5

Tivoli Policy Director for MQSeries, Version 3.8.1, supports the following IBM MQSeries family members:

- MQSeries, Version 5.1 (server only)
- MQSeries, Version 5.2 (server only)
- MQSeries, Version 5.2.1 (server only on Windows)
- MQSeries Integrator (MQSI), Version 2.02
- WebSphere MQ Integrator, Version 2.1
- MQSeries Workflow, Version 3.3

Tivoli Policy Director for MQSeries, Version 3.8.1, supports the following IBM MQSeries applications:

- BMC SoftwarePatrol for MQ
  - Patrol for MQ Optimizer, Versions 1.2.02 and 1.3.00
  - Patrol for MQ Operator, Version 2.2.01
  - Patrol for MQ Administrator, Version 3.1.01
- Tivoli Data Exchange, Version 1.2

Tivoli Policy Director for MQSeries, Version 3.8.1, supports the following application programming interfaces (APIs):

- Message queue interface (MQI)
- Java™ Message Service (JMS)

  For JMS, only bindings mode is supported.
- Application messaging interface (AMI)

New Features in Tivoli Policy Director for MQSeries

Tivoli Policy Director for MQSeries, Version 3.8.1, provides the following new features:

- Seamless integration and support of the following hardware accelerator cards:
  - nCipher nForce cards, model 300 SCSI card (Windows 2000 platform only)
  - Rainbow CS200 and CS600 cards

  For more information, see Appendix D, "Hardware Accelerator Support" on page 159.
- Support of MQSeries clusters

  For more information, see "Using Tivoli Policy Director for MQSeries with MQSeries Clusters" on page 64.

Tivoli Policy Director for MQSeries Environment

Figure 1 shows a block diagram of the core Tivoli Policy Director for MQSeries components and the security infrastructure components (in shaded areas).

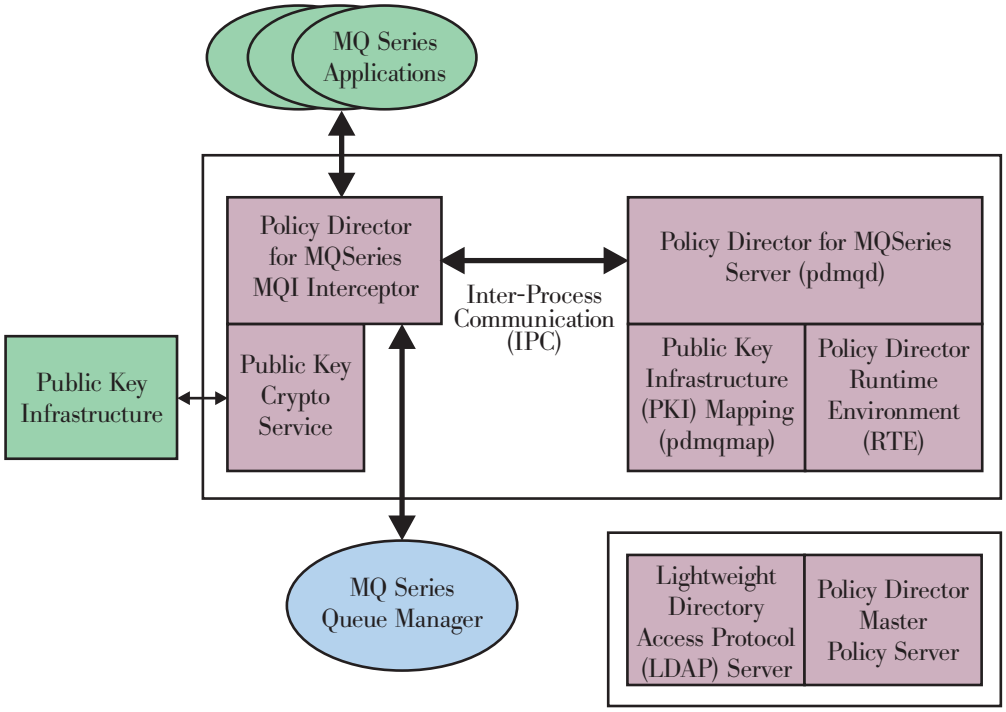

Figure 1. Tivoli Policy Director for MQSeries environment

Note: The infrastructure components do not necessarily reside in the same system as Tivoli Policy Director for MQSeries and MQSeries. However, Tivoli Policy Director for MQSeries needs to be able to access the services of all these components.

Components and Dependencies

This section has the following objectives:

- Provide background on the role played by each of the infrastructure components
- Refer to the documents that provide complete installation information for the infrastructure components
- Highlight some of the steps associated with the installation, initial setup, and configuration of the infrastructure and environment components

Tivoli Policy Director for MQSeries Libraries

The key piece of Tivoli Policy Director for MQSeries is a set of multithreaded, shared libraries that execute in the process space of an MQSeries application. The Tivoli Policy Director for MQSeries libraries intercept MQSeries API calls and enable MQSeries applications to be secured without any changes.

Lightweight Directory Access Protocol Directory

Tivoli Policy Director uses a Lightweight Directory Access Protocol (LDAP) server as its user registry. You must install and configure the LDAP directory server

before you install Tivoli Policy Director. Currently, Tivoli Policy Director supports the IBM SecureWay Directory and Netscape directory servers.

You can install the IBM SecureWay® Directory on a Solaris, AIX, Windows NT, or Windows 2000 system. Install the IBM SecureWay Directory on a machine designated as your official data repository. The IBM SecureWay Directory is included on the Tivoli Policy Director CDs that are part of the Tivoli Policy Director for MQSeries package.

Complete installation instructions are available in the following manuals:

- *IBM SecureWay Directory for the Solaris Operating Environment Software Installation and Configuration, Version 3.2*
- *IBM SecureWay Directory Installation and Configuration for AIX*
- *IBM SecureWay Directory Installation and Configuration for Windows*

Refer to the installation manual that is appropriate to your platform.

Tivoli Policy Director

Tivoli Policy Director for MQSeries supports Tivoli Policy Director versions 3.7.1 and 3.8 servers and clients, and it relies on Tivoli Policy Director for the following services:

- Enterprise user registry for Tivoli Policy Director users
- Centralized system to define authorization and data protection policy for access to MQSeries resources, such as queues

Tivoli Policy Director for MQSeries uses the industry standard authorization API to obtain data protection and authorization policy from the master policy server.

Tivoli Policy Director can reside on a Solaris, AIX, Windows NT, Windows 2000, Linux, or HP/UX system. Refer to the installation instructions for the platform that hosts the Tivoli Policy Director product.

Other required information is provided in the following manuals:

- *Tivoli SecureWay Policy Director Base Installation Guide*
- *Tivoli SecureWay Policy Director Base Administration Guide*

Tivoli Policy Director for MQSeries depends on the following components of Tivoli Policy Director:

- Tivoli Policy Director runtime environment must be installed on each machine that is running Tivoli Policy Director for MQSeries.
- Tivoli Policy Director master policy server must be installed one time in the Tivoli Policy Director for MQSeries environment.
- Tivoli Policy Director Web Portal Manager is optional, and if used, it must be installed one time in the Tivoli Policy Director for MQSeries environment.

Public Key Infrastructure

Tivoli Public Key Infrastructure runtime services are provided by the GSKIT component of Tivoli Policy Director for MQSeries. GSKIT allows the user to request certificates, store certificates, and apply keys to provide data integrity and data privacy. GSKIT can use certificates issued by any of the following certificate authority products:

- Tivoli Public Key Infrastructure, Version 3.7.1

- Entrust WebConnector, Version 5.0
- iPlanet Certificate Management System (CMS), Version 4.2
- PKIServ application for RACF on OS/390, Release 10
- Baltimore UniCERT, Version 3.5

Chapter 2. Installation and Uninstallation of Tivoli Policy Director for MQSeries

This chapter describes the installation and uninstallation of Tivoli Policy Director for MQSeries, Version 3.8.1, so that it conforms to the model used by Tivoli Policy Director. Tivoli Policy Director for MQSeries is delivered on the following platforms: Solaris 7 and 8, AIX 4.3.3, Windows NT 4.0, Service Pack 6 and above, and Windows 2000, Service Pack 2.

This chapter consists of the following sections:

Installation Prerequisites

Before installing Tivoli Policy Director for MQSeries, Version 3.8.1, the following software must be installed and configured in your environment. You cannot install Tivoli Policy Director for MQSeries if these requirements are not met.

Environment Prerequisites for Tivoli Policy Director for MQSeries

Tivoli Policy Director for MQSeries must have access to each of the following:

- Tivoli Policy Director master policy server
- Tivoli Policy Directory user registry —Tivoli Policy Director maintains its users and groups on a user registry that is based in Lightweight Directory Access Protocol (LDAP). You should install and configure an LDAP server for Tivoli Policy Director usage (schema loading) before installing Tivoli Policy Director.
- Tivoli Policy Director Web Portal Manager (optional).

Note: These products do not necessarily have to be installed on the same machine.

Local Prerequisites for Tivoli Policy Director for MQSeries

Each machine must have the following software installed:

- Tivoli Policy Director runtime environment, a component of Tivoli Policy Director.
- MQSeries server.

- For Windows NT platforms only, Java Runtime Environment must be in the path for you to use the interactive logon. You can use the Java Runtime Environment shipped with the LDAP client. The following path is automatically set in your environment:

  \Program Files\IBM\LDAP\jre\bin
- The LDAP client package of the IBM SecureWay LDAP Directory.

You can find the IBM SecureWay LDAP client on the *Tivoli Policy Director Base for Sun Solaris* CD, the *Tivoli Policy Director Base for AIX* CD, or the *Tivoli Policy Director Base for Windows* CD (for Windows platforms).

To install the LDAP client from the *Tivoli Policy Director Base for Sun Solaris* CD, perform the following steps:

1. Insert the *Tivoli Policy Director Base for Sun Solaris* CD into the CD-ROM drive.
2. Mount it to /cdrom.
3. Run the following command:

   ```
   $ pkgadd -d /cdrom/SecureWay_Directory/ldap32_us
   IBMldapc
   ```

To install the LDAP client from the *Tivoli Policy Director Base for AIX* CD, perform the following steps:

1. Insert the *Tivoli Policy Director Base for AIX* CD into the CD-ROM drive.
2. Mount it to /cdrom.
3. Start the System Management Interface Tool (SMIT). The **System Management** menu displays.
4. Click **Software Installation and Maintenance**.
5. Click **Install and Update Software**.
6. Click **Install and Update Software from Latest Available Software**.
7. Specify the directory from which you are installing the software. For example, if the CD is mounted on /cdrom, enter the following path:

   ```
   /cdrom/SecureWay_Directory/ldap32_us
   ```
8. Click **OK**.
9. Click **List** to display the list of software that can be installed.
10. Select **ldap.client** and click **OK**.
11. Accept the defaults for all other data fields and click **OK**.
12. Click **OK** in the dialog box that displays the message:

    ```
    !Continuing may delete information you may want to keep.
    This is your last chance to stop before continuing.
    ```
13. When the installation process is complete, click **DONE**.

To install the LDAP client from the *Tivoli Policy Director Base for Windows* CD, perform the following steps:

1. Insert the *Tivoli Policy Director Base for Windows CD* into the CD-ROM drive.
2. Use Windows Explorer or My Computer to find the \SecureWay_Directory\ldap32_us directory on the CD.
3. Double-click **setup.exe**.

Installing and Uninstalling on a Solaris Platform

This section contains instructions for installing and uninstalling Tivoli Policy Director for MQSeries, Version 3.8.1, on a Solaris platform.

Installing Tivoli Policy Director for MQSeries on Solaris

Tivoli Policy Director for MQSeries requires that infrastructure components are installed and operational for it to function properly.

Perform the following steps to install Tivoli Policy Director for MQSeries on a Solaris 7 or 8 platform:

1. Stop the MQSeries server.
2. Log on as root.
3. Insert the Tivoli Policy Director for MQSeries CD into the CD-ROM drive and mount it to /cdrom.
4. Run the following commands:

```
$ pkgadd -d /cdrom/cdrom0/Solaris/ gsk5bas
$ pkgadd -d /cdrom/cdrom0/Solaris/ PDMQrte
```

5. Answer `Y` at the prompt ″Do you want to continue with the installation?″
6. The installation of Tivoli Policy Director for MQSeries creates the following directory tree:

```
/opt/pdmq/.configure
/opt/pdmq/bin
/opt/pdmq/doc
/opt/pdmq/etc
/opt/pdmq/lib
/opt/pdmq/nls
/opt/pdmq/nls/msg
/var/pdmq/audit
/var/pdmq/db
/var/pdmq/keytab
/var/pdmq/log
```

Uninstalling Tivoli Policy Director for MQSeries on Solaris

Perform the following steps to uninstall Tivoli Policy Director for MQSeries on a Solaris 7 or 8 platform:

1. Log on as root.
2. Stop the MQSeries server.
3. Unconfigure Tivoli Policy Director for MQSeries. For more information, see “Performing a Complete Unconfiguration of Tivoli Policy Director for MQSeries” on page 19.
4. Run the following commands:

```
$ pkgrm PDMQrte
$ pkgrm gsk5bas
```

Installation and Uninstallation on an AIX Platform

This section contains instructions for installing and uninstalling Tivoli Policy Director for MQSeries, Version 3.8.1, on an AIX platform.

Installing Tivoli Policy Director for MQSeries on AIX

Tivoli Policy Director for MQSeries requires that infrastructure components are installed and operational for it to function properly.

Perform the following steps to install Tivoli Policy Director for MQSeries on AIX:

1. Stop the MQSeries server.
2. Log on as root, insert the Tivoli Policy Director for MQSeries CD into the CD-ROM drive, and mount it to /cdrom.
3. Start SMIT. The **System Management** menu is displayed.
4. Click **Software Installation and Maintenance**.
5. Click **Install and Update Software**.
6. Click **Install and Update Software from Latest Available Software**.
7. Enter the directory from which you are installing the software. For example, if the CD is mounted on /cdrom, enter the following path:

   ```
   /cdrom/AIX
   ```
8. Click **OK**.
9. Click **List** to display a list of software that you can install.
10. Select both **Policy Director for MQSeries** and **gskkm**.
11. Type `no` in the **PreviewOnly** option.
12. Click **OK**.
13. Accept the default settings for the remaining options and click **OK**.
14. Click **OK** in the dialog box that displays the following message:

    ```
    !Continuing may delete information you may want to keep.
     This is your last chance to stop before continuing.
    ```
15. When the installation process is complete, click **DONE**.
16. Tivoli Policy Director for MQSeries creates the following directory tree:

    ```
    /opt/pdmq
    /opt/pdmq/.configure
    /opt/pdmq/bin
    /opt/pdmq/doc
    /opt/pdmq/etc
    /opt/pdmq/lib
    /opt/pdmq/nls
    /opt/pdmq/nls/msg
    /var/pdmq/audit
    /var/pdmq/db
    /var/pdmq/keytab
    /var/pdmq/log
    ```

Uninstalling Tivoli Policy Director for MQSeries on AIX

Perform the following steps to uninstall Tivoli Policy Director for MQSeries on an AIX platform:

1. Unconfigure Tivoli Policy Director for MQSeries. For more information, see "Performing a Complete Unconfiguration of Tivoli Policy Director for MQSeries" on page 19.
2. Stop the MQSeries server.
3. Log on as root.
4. Start SMIT. The **System Management** menu is displayed.
5. Click **Software Installation and Maintenance**.
6. Click **Software Maintenance and Utilities**.
7. Click **Remove Installed Software**.
8. Click **List**.
9. In the **Multi-select List**, select **PDMQ.rte** and **gskkm.rte**.
10. Type `no` in the **PREVIEW only?** text field.

11. Click **OK** in the dialog box that displays the following message:

```
!Continuing may delete information you may want to keep.
This is your last chance to stop before continuing.
```

12. When the uninstallation is complete, click **Done**.

Installation and Uninstallation on a Windows NT or Windows 2000 Platform

This section contains instructions for installing and uninstalling Tivoli Policy Director for MQSeries, Version 3.8.1, on a Windows NT or Windows 2000 platform.

Installing Tivoli Policy Director for MQSeries on Windows

Tivoli Policy Director for MQSeries requires that infrastructure components are installed and operational for it to function properly.

Perform the following steps to install Tivoli Policy Director for MQSeries on a Windows platform:

1. Log on as Administrator.
2. Stop the MQSeries server.
3. Insert the Tivoli Policy Director for MQSeries CD into the CD-ROM drive.
4. If the Choose Setup Language window is displayed, the **setup** program has run automatically. Go to Step 7. If not, continue to Step 5.
5. Change the directory to \WinNT.
6. Run **setup**.
7. In the Choose Setup Language window, choose the appropriate language.
8. Click **OK**.
9. In the Tivoli Policy Director for MQSeries Setup (Welcome to the InstallShield Wizard) window, click **Next**.
10. In the Tivoli Policy Director for MQSeries Setup (License Agreement) window, read the license agreement and click **Yes**.
11. In the Tivoli Policy Director for MQSeries Setup (Choose Destination Location) window, click **Browse**.
12. In the Choose Folder window, select the folder and path where you want to install Tivoli Policy Director for MQSeries and click **OK**.
13. In the Tivoli Policy Director for MQSeries Setup (Choose Destination Location) window, verify that the folder displayed is correct and click **Next**.
14. When the Tivoli Policy Director for MQSeries Setup (InstallShield Wizard Complete) window displays, click **Finish**.

    Note: Your system will reboot automatically.

Note: The following directories are created:

pdmq_install_path\.configure
pdmq_install_path\audit
pdmq_install_path\bin
pdmq_install_path\db
pdmq_install_path\doc
pdmq_install_path\etc
pdmq_install_path\keytab
pdmq_install_path\lib

```
pdmq_install_path\log
pdmq_install_path\nls
pdmq_install_path\nls\msg
pdmq_install_path\classes
```

By default *pdmq_install_path* is C:\Program Files\Tivoli\Policy Director for MQSeries. You have the option to specify a different location for your installation base.

Verifying File Permissions for the MQSeries Daemon Service

Before configuring Tivoli Policy Director for MQSeries, you must verify that the Tivoli Policy Director daemon service has full control of the *pdmq_install_path*\audit and *pdmq_install_path*\db directories. Also, users who belong to the mqm group must have full control of the *pdmq_install_path*\log directory. Complete the following steps to set the permissions for the audit, db, and log directories:

1. Start the Windows NT or Windows 2000 Explorer.
2. Right-click the name of the *pdmq_install_path*\audit directory to display the pop-up menu.
3. Click **Properties** to display the **audit Properties** dialog box.
4. Click the **Security** tab.
5. Click **Permissions** to display the **Directory Permissions** window.
6. Review the **Name** box. If **Everyone** is not assigned **Full Control**, continue to the next step. If **Everyone** is assigned **Full Control**, you do not need to make any changes in the permissions for this directory and you can go to step 12.
7. Click **Add** to display the **Add Users and Groups** window.
8. Select **MUSR_PDMQD**.
9. Click **Add** to display the selected user in the **Add Names** field.
10. Click **OK**.
11. Ensure that **Full Control** is displayed in the **Type of Access** field.
12. Click **OK** until you return to the Windows Explorer.
13. If you have not set the permissions for the *pdmq_install_path*\db directory, repeat step 1 through step 12, but select the *pdmq_install_path*\db directory in step 2.
14. If you have not set the permissions for the *pdmq_install_path*\log directory, repeat step 2 through step 12, but select the *pdmq_install_path*\log directory in step 2 and select the group **mqm** in Step 8.

Uninstalling Tivoli Policy Director for MQSeries on Windows

Perform the following steps to uninstall Tivoli Policy Director for MQSeries on a Windows platform:

1. Use the instructions under "Using the Wizard to Perform Complete Unconfiguration of Tivoli Policy Director for MQSeries" on page 25 to unconfigure Tivoli Policy Director for MQSeries.
2. Click **Start** → **Settings** → **Control Panel** → **Add/Remove Programs**.
3. From the list of programs displayed in the Add/Remove Programs window, select **Tivoli Policy Director for MQSeries**.
4. Click the **Change/Remove** button in Windows 2000 or the **Add/Remove** button in Windows NT.
5. In the Choose Setup Language window, choose the appropriate language from the drop-down list.

6. Click **OK**.
7. From the Confirm File Deletion window, click **OK**.

   Note: Your machine automatically reboots.

Creating the secPKIMap Object Class in LDAP

Tivoli Policy Director for MQSeries requires that LDAP secMap object be extended using an auxiliary object **secPKIMap**. For more information, see "Mapping Public Key Infrastructure Identities to Tivoli Policy Director Users" on page 55.

Before you configure Tivoli Policy Director for MQSeries, Version 3.8.1, you must create the **secPKIMap** object class in LDAP by using the **ldapmodify** command, which passes the pdmq.ldif file as input. The pdmq.ldif file is located in the /etc directory of the Tivoli Policy Director for MQSeries.

To create the **secPKIMap** object class, run the following command from the MS-DOS command prompt:

```
ldapmodify -h hostname -p 389 -D admin_ID -w password -f
\pdmq_install_path\etc\pdmq.ldif
```

where:

hostname
: The host name of the machine being configured.

389
: The default LDAP port number.

admin_ID
: The LDAP administrator ID specified earlier.

password
: The LDAP administrator password specified earlier.

Chapter 3. Configuration

Before starting configuration, you must restart your Tivoli Policy Director server and LDAP server if they reside on the same machine as your Tivoli Policy Director for MQSeries server. All MQSeries processes must be stopped before you start Tivoli Policy Director for MQSeries configuration.

If you are using a Solaris or AIX platform, you must use command line utilities to configure and unconfigure Tivoli Policy Director for MQSeries, Version 3.8.1. If you are using a Windows NT or Windows 2000 platform, you must use the graphical configuration wizard. To use command line utilities, use the instructions in the "Using Command Line Utilities to Perform Configuration and Unconfiguration Tasks" section of this chapter. To use the Windows graphical configuration wizard, use the instructions in the "Using the Windows Graphical Configuration Wizard to Configure and Unconfigure Tivoli Policy Director for MQSeries" on page 20.

Validating the MQSeries Environment

As a first task, administrators need to configure and set up the MQSeries environment for the secure domain being protected under Tivoli Policy Director for MQSeries. This task includes defining MQSeries objects, such as queue managers, channels, and local, remote, model, alias, and transmission queues. Also, validation includes testing communication between MQSeries servers to be sure it is working properly.

In installations where MQSeries is already functioning, this initial task is done before you install Tivoli Policy Director for MQSeries. Refer to the *MQSeries Planning Guide* and the *MQSeries System Administration Manual* for assistance in defining the MQSeries objects.

Identifying the MQSeries Resources to Be Protected

After all MQSeries resources are defined and operational, the next step is to populate these resources in the Tivoli Policy Director–protected object name space. The objects that appear in this hierarchical name space represent the actual protected resources. Tivoli Policy Director expects policy and access control templates attached to these resources. The Tivoli Policy Director for MQSeries configuration command initially creates the required protected object space.

Using Command Line Utilities to Perform Configuration and Unconfiguration Tasks

Configuration of Tivoli Policy Director for MQSeries, Version 3.8.1, on UNIX requires the use of only one command, **pdmqcfg**. For unconfiguration, the only command required is **pdmqucfg**. You must issue these commands as root on either the Solaris or AIX platform, and you must have the appropriate authority to do so.

The configuration command **pdmqcfg** has the following options:

–all Performs the initial full configuration of Tivoli Policy Director for MQSeries.

–add Adds a new queue manager to the Tivoli Policy Director for MQSeries protected object space.

–update
Updates a queue manager entry in the Tivoli Policy Director for MQSeries protected object space.

–enable
Enables Tivoli Policy Director for MQSeries interception of MQSeries applications.

–? Displays syntax information for the **pdmqcfg** command.

The unconfiguration command **pdmqucfg** has the following options:

–all Unconfigures Tivoli Policy Director for MQSeries completely.

–delete
Removes a queue manager from the Tivoli Policy Director for MQSeries protected object space.

–disable
Disables Tivoli Policy Director for MQSeries interception of MQSeries applications.

–? Displays syntax information for the **pdmqucfg** command.

This section contains the instructions for using **pdmqcfg** and **pdmqucfg** for the following tasks:

Note: If any command documented in this section produces an error message, see the configuration log /var/pdmq/log/config.log or Appendix B, "Problem Determination" on page 121 for more information.

Performing the Initial Complete Configuration of Tivoli Policy Director for MQSeries

You must run the **pdmqcfg** command as root on Solaris or AIX platforms. On Windows platforms, you must run the pdmqcfg.exe program as Administrator. All MQSeries processes must be stopped before performing this operation.

The complete configuration includes the following tasks:

1. Creating the /PDMQ/Queue entry in the Tivoli Policy Director protected object space.
2. Creating a group named pdmq-admin in the Tivoli Policy Director protected object space.

Note: A user must belong to the pdmq-admin group in order to browse or get messages from the error handling queue.

3. Creating a new protected object policy (POP) named pdmq-erq-def-pop, which is attached to the error queue when the error queue is created.
4. Creating a new access control list (ACL) called pdmq-erq-def-acl, which is attached to the error queue when the error queue is created.
5. Enabling the secure sockets layer (SSL) connection with the policy server.
6. Enabling the Tivoli Policy Director for MQSeries MQI library interceptor and server (daemon).
7. Adding the queue manager and the queues to the Tivoli Policy Director protected object space.

   In the Tivoli Policy Director protected object space, this step creates a single entry for the queue manager and an entry for each queue (with the exception of SYSTEM and transmission queues). For example, if queue manager AUSTIN hosts the queues CACTUS and WATERLOO, after running this step, the protected object space would contain the following entries:

   ```
   /PDMQ/Queue/AUSTIN/CACTUS
   /PDMQ/Queue/AUSTIN/WATERLOO
   ```

8. Creating an error handling queue in the Tivoli Policy Director protected object space, where pdmq-erq-def-pop and pdmq-erq-def-acl are then attached to it.

   Note: If the name of the error handling queue is not specified, the default name DEAD.QUEUE is used. The error handling queue is created in the MQSeries queue manager if it does not exist already.

Use the following command syntax to perform the initial configuration:

```
pdmqcfg -all -admin admin_ID -password password -qm queue_manager
[-errorq error_queue_name]
```

Note: You should run the **pdmqcfg** command with the **–all** option only once—for the initial configuration.

where:

admin_ID
: Your Tivoli Policy Director administrator ID.

password
: Your Tivoli Policy Director administrator ID password.

queue_manager
: The name of the specific queue manager entry you need to export to the Tivoli Policy Director object space.

error_queue_name
: The name of the error handling queue, which is assigned to handle invalid messages associated with the specified queue manager.

  Note: Tivoli Policy Director for MQSeries creates the error handling queue if it does not exist.

If the configuration fails, refer to Appendix B, correct the problem, and run the **pdmqcfg** command again. See Appendix B, “Problem Determination” on page 121 for more information.

Enabling Message Queue Interface Library Interception

Before you enable the message queue interface (MQI) library interception, you must stop all MQSeries processes. Use the following command syntax to enable the MQI library interception and server (daemon):

```
pdmqcfg -enable
```

When you issue this command from a system, Tivoli Policy Director for MQSeries begins protecting MQSeries resources on this system.

Adding a New Queue Manager to the Protected Object Space

Use the following command syntax to add a new queue manager to the Tivoli Policy Director protected object space:

```
pdmqcfg -add -admin admin_ID -password password -qm queue_manager
[-errorq error_queue_name]
```

where:

admin_ID
: Your Tivoli Policy Director administrator ID.

password
: Your Tivoli Policy Director administrator ID password.

queue_manager
: The name of the specific queue manager you need to export to the Tivoli Policy Director object space. All the queues that belong to this *queue_manager* will be added to the Tivoli Policy Director object space except system and transmit queues.

error_queue_name
: The name of the specific queue that is assigned to handle invalid messages associated with the specified queue manager. This queue is created in MQSeries if it does not exist. If you do not specify an error queue, Tivoli Policy Director for MQSeries creates the queue DEAD.QUEUE and specifies it as the error queue.

Updating an Existing Queue Manager Definition

For updating an existing queue manager to the Tivoli Policy Director protected object space, the command syntax is:

```
pdmqcfg -update -admin admin_ID -password password -qm queue_manager
   [-errorq error_queue_name]
```

where:

admin_ID
: Your Tivoli Policy Director administrator ID.

password
: Your Tivoli Policy Director administrator ID password.

queue_manager
: The name of the specific queue manager to update in the Tivoli Policy Director object space.

error_queue_name
: The name of the specific queue assigned to handle invalid messages associated with the specified queue manager.

Note: If the specified error queue does not exist, Tivoli Policy Director for MQSeries creates it. If the **errorq** parameter is specified, then the **errorq** value for the queue manager is also updated.

This command refreshes the list of queues for the specified queue manager.

Displaying Configuration Help Information

Use the following command syntax to display configuration help:

```
pdmqcfg -?
```

Performing a Complete Unconfiguration of Tivoli Policy Director for MQSeries

The complete unconfiguration process for Tivoli Policy Director for MQSeries performs the following tasks:

- Removing all local queue managers and queues which are configured in the Tivoli Policy Director protected object space.
- Disabling the Tivoli Policy Director for MQSeries library interceptor and server (daemon).
- Disabling the SSL connection with the policy server.
- Removing the /PDMQ/Queue entry from the Tivoli Policy Director protected object space if there are no queue managers configured under this entry.

Use the following command syntax to completely unconfigure Policy Director for MQSeries:

```
pdmqucfg -all -admin admin_ID -password password [-force]
```

where:

admin_ID
Your Tivoli Policy Director administrator ID.

password
Your Tivoli Policy Director administrator ID password.

–force Disables confirmation prompts.

Disabling Message Queue Interface Library Interception

Use the following command syntax to disable message queue interface (MQI) library interception and discontinue providing Tivoli Policy Director for MQSeries security on this system. All MQSeries processes must be stopped before performing this operation.

```
pdmqucfg -disable
```

Removing a Queue Manager from the Protected Object Space

Use the following command syntax to remove a queue manager from the Tivoli Policy Director protected object space:

```
pdmqucfg -delete -qm queue_manager -admin admin_ID -password password [-force]
```

where:

admin_ID
Your Tivoli Policy Director administrator ID.

password
Your Tivoli Policy Director administrator ID password.

queue_manager
The name of the specific queue manager you need to remove from the Tivoli Policy Director object space.

–force Disables confirmation prompts.

This command removes an individual queue manager entry from the Tivoli Policy Director protected object space, but it does not delete the error queue from the local queue manager in MQSeries.

Displaying Unconfiguration Help

Use the following command syntax to display unconfiguration help:

```
pdmqucfg -?
```

Using the Windows Graphical Configuration Wizard to Configure and Unconfigure Tivoli Policy Director for MQSeries

Note: If you are using a Solaris or AIX platform, you must use the instructions for the command line utilities under "Using Command Line Utilities to Perform Configuration and Unconfiguration Tasks" on page 15 in this chapter.

If you install Tivoli Policy Director for MQSeries on a Windows platform, you can perform any of the following tasks using the graphical configuration wizard:

- "Using the Wizard to Perform Initial Complete Configuration"
- "Using the Wizard to Enable or Disable Message Queue Interface Library Interception" on page 23
- "Using the Wizard to Add, Update, or Delete Queue Managers" on page 23
- "Using the Wizard to Perform Complete Unconfiguration of Tivoli Policy Director for MQSeries" on page 25

Note: If you receive an error message while performing any of these tasks using the Wizard, click **Show Log** to view the configuration log.

Using the Wizard to Perform Initial Complete Configuration

Complete configuration of Tivoli Policy Director for MQSeries involves several tasks, as explained in a previous section. For details, See "Performing the Initial Complete Configuration of Tivoli Policy Director for MQSeries" on page 16. To use the Wizard to perform initial configuration of Policy Director for MQSeries, use the following steps:

1. Click the menu path **Start → Programs → Policy Director for MQSeries → pdmqcfg**. The Tivoli Policy Director for MQSeries window displays as shown in Figure 2 on page 21.

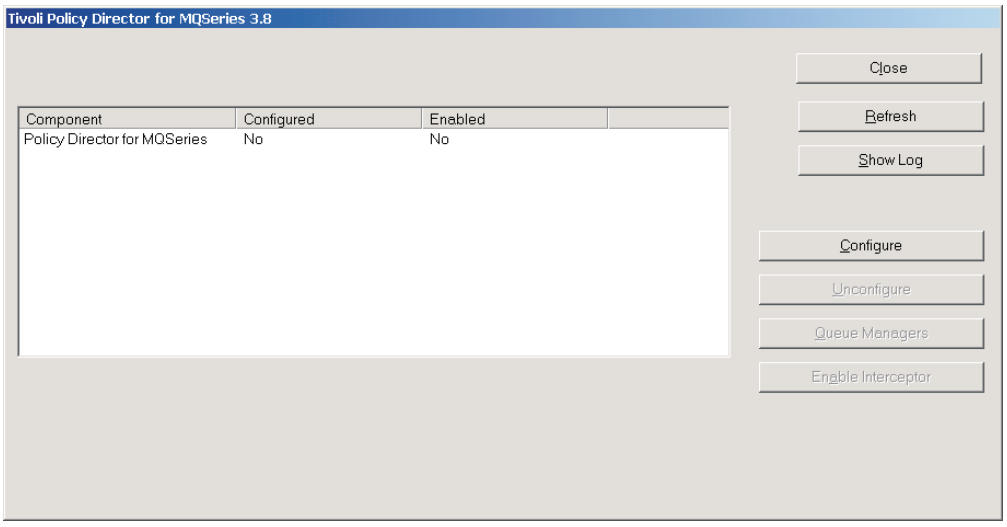

Figure 2. Initial Windows graphical configuration wizard screen

Note: Because this is the first time you use **pdmqcfg.exe**, the **Configure** button is enabled. Also, because you have not yet configured Tivoli Policy Director for MQSeries, the word **No** is displayed under **Configured** and the word **No** is displayed under **Enabled** in the box.

2. Click **Configure**. The Initial Policy Director for MQSeries Configuration dialog box displays, as shown in Figure 3 on page 22.

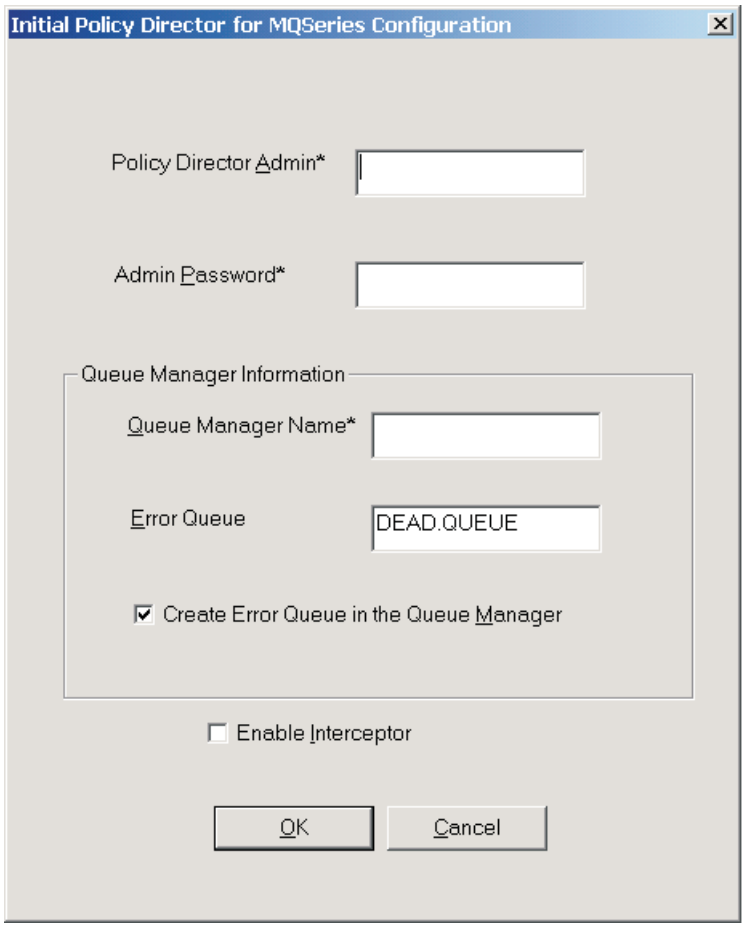

Figure 3. Initial configuration dialog box

3. Complete the following fields:
   - **Policy Director Admin** – your Tivoli Policy Director administrator ID.
   - **Admin Password** – your Tivoli Policy Director administrator ID password.
   - **Queue Manager Name** – the queue manager.

     Note: To add more queue managers later, you can click **Queue Managers** in the Tivoli Policy Director for MQSeries window. This button is activated once the configuration is complete.
   - **Error Queue** the error handling queue.

     Note: The default error queue name is **DEAD.QUEUE**. The error handling queue is created in the MQSeries queue manager if it doesn't exist already.
4. If you choose to create the error queue in MQSeries during configuration, check the box to the left of **Create Error Queue in the Queue Manager**.
5. If you choose to enable the Tivoli Policy Director for MQSeries interceptor here, check the box to the left of **Enable Interceptor**.

   Note: If you do not enable the interceptor at this time, you can click **Enable Interceptor** in the Tivoli Policy Director for MQSeries window at a later time. This button is activated once the configuration is complete.

When the configuration is complete, the configuration status changes to **Yes**.

Using the Wizard to Enable or Disable Message Queue Interface Library Interception

Perform the following steps to enable or disable MQI library interception and the Tivoli Policy Director for MQSeries Server (daemon):

1. Click the menu path **Start** → **Programs** → **Policy Director for MQSeries** → **pdmqcfg** to open the Tivoli Policy Director for MQSeries window. See Figure 4.

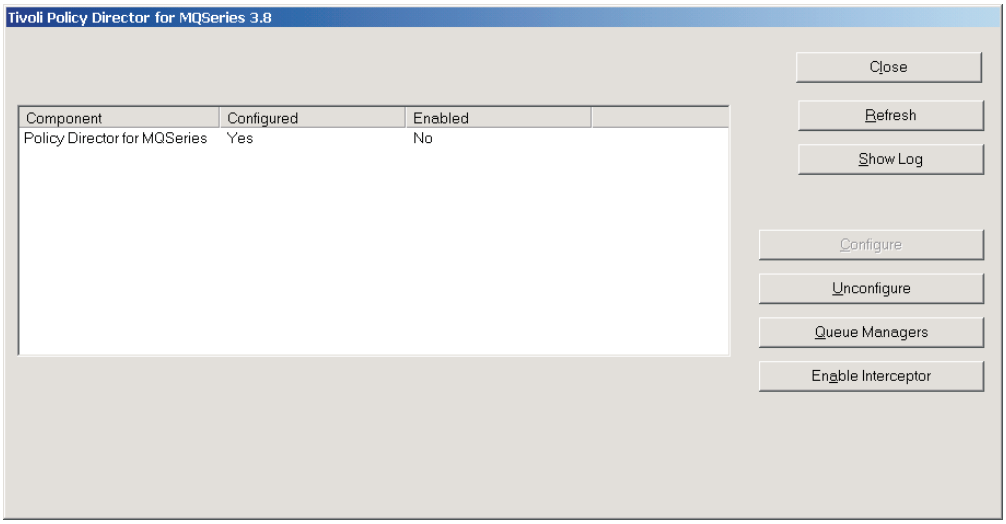

Figure 4. Wizard window after initial configuration is complete

Note: If the **Enable Interceptor** button is not activated, then the initial configuration is not successful. For further instructions, return to "Using the Wizard to Perform Initial Complete Configuration" on page 20.

2. Click **Enable Interceptor**. If the MQI library interceptor was not enabled, clicking the button enables it. If it was enabled, clicking the button **Disable Interceptor** disables it.
3. Review the box in the Tivoli Policy Director for MQSeries window to see if the appropriate indicator, **Yes** (enabled) or **No** (disabled), is displayed under **Enabled**.

Using the Wizard to Add, Update, or Delete Queue Managers

Perform the following steps to add or remove a queue manager from Tivoli Policy Director for MQSeries:

1. Click the menu path **Start** → **Programs** → **Policy Director for MQSeries** → **pdmqcfg** to open the Tivoli Policy Director for MQSeries window.
2. Enter your administrative user ID and password.
3. Click **Queue Managers** to open the Queue Manager Dialog dialog box, which is shown in Figure 5 on page 24. The list of queue managers currently in the Policy Director object space displays in the **List of added Queue Managers** box. (In the figure shown, this box is empty, so no queue managers have been created.)

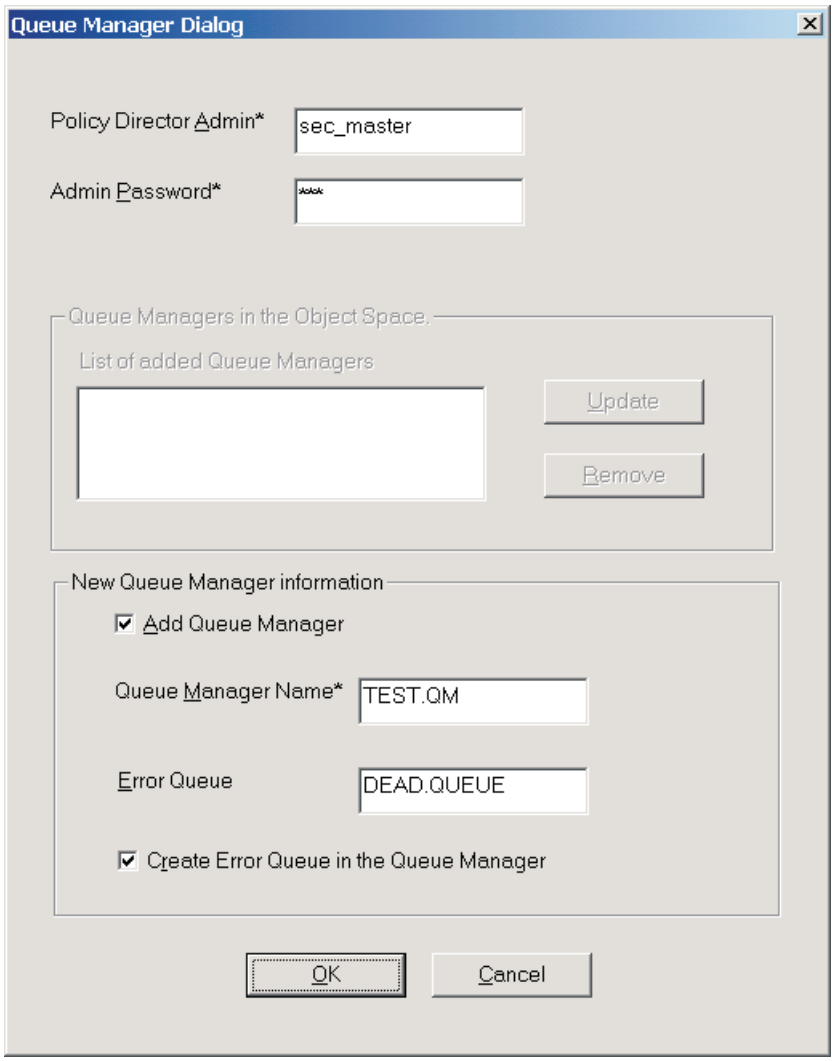

Figure 5. Queue Manager Dialog dialog box (no queue managers)

4. To add a queue manager, check the box to the left of **Add Queue Manager** and enter the name of the queue manager in the **Queue Manager Name** field.
5. Click **OK**. The queue manager you added is now displayed in the **List of added Queue Managers** box, as shown in Figure 6 on page 25.

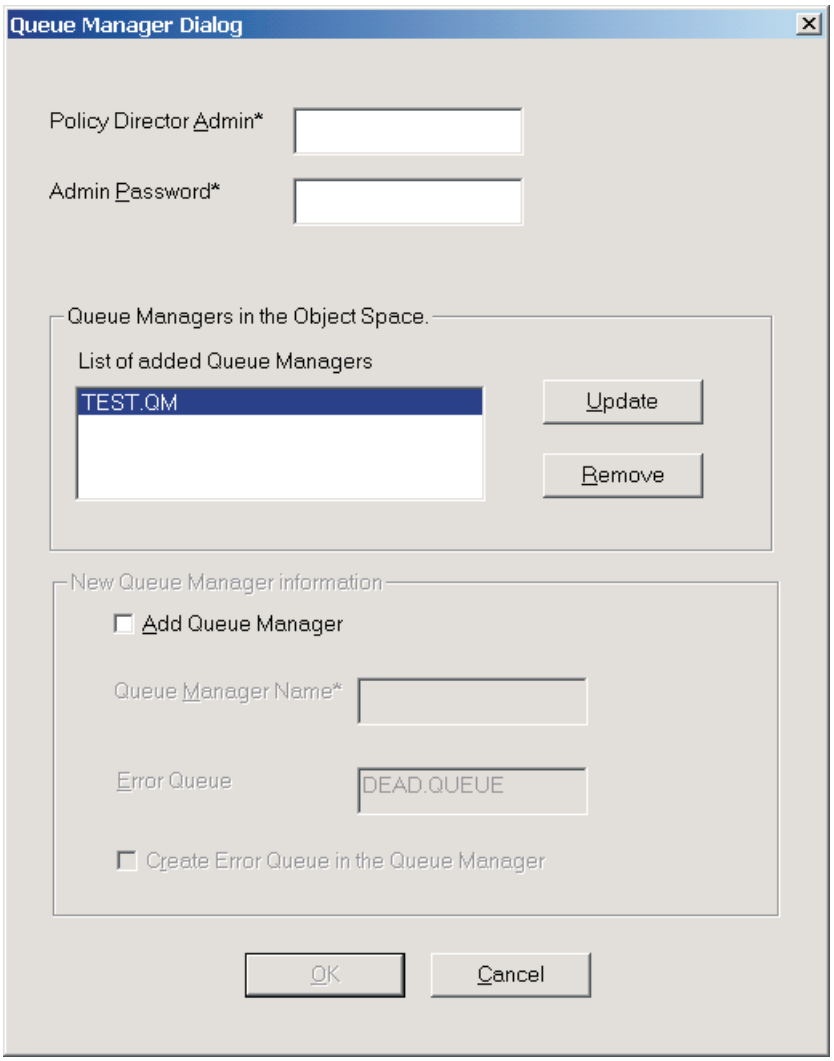

Figure 6. Queue Manager Dialog dialog box (queue manager added)

6. If you need to create an error queue for the specified queue manager, enter the name in the **Error Queue** field.

   Note: If you do not enter an error handling queue name, the default name **DEAD.QUEUE** is used.
7. If you need the error queue to be created in MQSeries for the specified queue manager, check the box next to **Create Error Queue in the Queue Manager** as shown in Figure 5 on page 24.
8. If you need to remove a queue manager from the existing list, select the name of the queue manager you need to remove and click **Remove**.
9. If you need to update a queue manager in the existing list, select the name of the queue manager you need to update and click **Update** to update all the queues in the object space.

Using the Wizard to Perform Complete Unconfiguration of Tivoli Policy Director for MQSeries

Completely unconfiguring Tivoli Policy Director for MQSeries performs the following tasks:

1. Removes all local queue managers and queues that are configured in the Tivoli Policy Director for MQSeries protected object space
2. Disables the MQI library interceptor and the Tivoli Policy Director for MQSeries server (daemon)
3. Disables the SSL connection with the policy server
4. Removes the /PDMQ/Queue entry from the Tivoli Policy Director protected object space if there are no queue managers configured under this entry.

After Tivoli Policy Director for MQSeries is configured, the **Unconfigure** button is activated in the Tivoli Policy Director for MQSeries window. From this window, perform the following steps to unconfigure Tivoli Policy Director for MQSeries:

1. Click **Unconfigure** to open the Policy Director for MQSeries Unconfiguration window, which is shown in Figure 7.

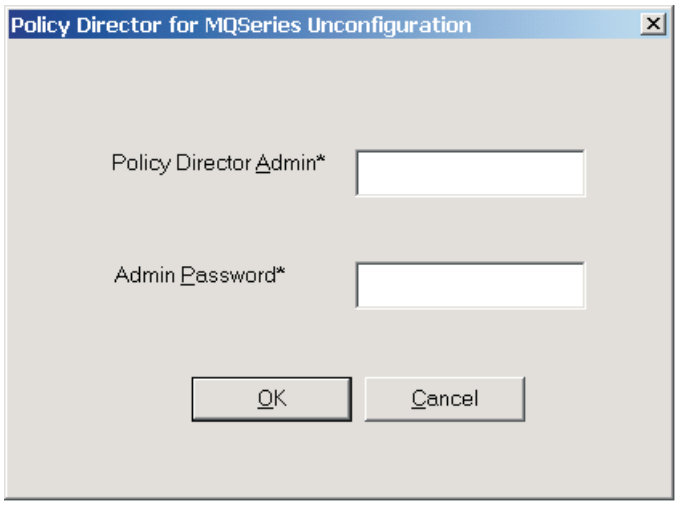

Figure 7. Windows graphical wizard unconfiguration window

2. Enter your administrator ID and password.
3. Click **OK**.
4. When the message `Do you really want to unconfigure Policy Director for MQSeries? [y/n]is` displayed, click **Yes**. The Question Dialog dialog box is shown in Figure 8.

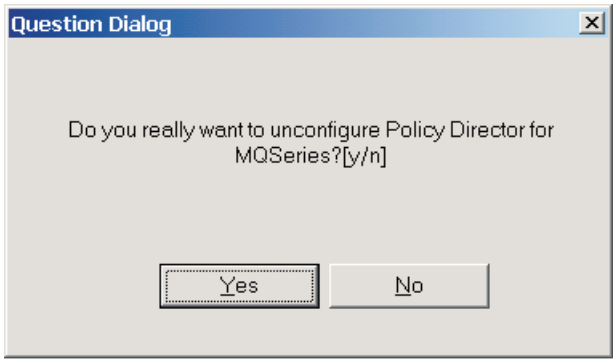

Figure 8. Windows graphical wizard unconfiguration confirmation window

Advanced Configuration

An advanced user may want to modify the Tivoli Policy Director for MQSeries configuration file that enables the authorization API connection with the policy server. This file namedpdmqazn.conf is located in *pdmq_install_path*/etc.

LDAP Server Using SSL

You can specify that Tivoli Policy Director for MQSeries uses the authorization API to communicate with the LDAP server using SSL. To do this, you must set additional parameters in the configuration file. For more information about these parameters, see the documentation in the pdmqazn.conf file. The required parameters in the [ldap] stanza of the pdmqazn.conf file are as follows:

ssl-enabled
(optional) Enables SSL communication with the LDAP server. Specify yes or no.

ssl-port
The SSL IP port on which the LDAP server listens for SSL requests. Required if ssl-enabled is specified as yes (default is 636).

ssl-keyfile
Path to an SSL key file containing the LDAP server certificate. Required if ssl-enable is specified as yes.

ssl-keyfile-pwd
Password for the key file. Required if ssl-enable is specified as yes unless a stash file is used.

ssl-keyfile-dn
Key file distinguished name (required for files with multiple keys).

The following is an example of the [ldap] stanza in the pdmqazn.conf file after the LDAP server using SSL is configured:

```
[ldap]
enabled=yes
host=hosta.austin.tivoli.com
port=389
bind-dn=cn=pdmqazn/hostb.austin.tivoli.com,
 cn=SecurityDaemons,secAuthority=Default
bind-pwd=pdmqazn
cache-enabled=true
#max-search-size=2048
#prefer-readwrite-server=true
#auth-using-compare=true
ssl-enabled=yes
ssl-port=636
ssl-keyfile=/tpki4.kdb
ssl-keyfile-pwd=pdsrv
ssl-keyfile-dn=cerdn
```

Tivoli Policy Director for MQSeries Cache Interval

The **pdmq-cache-interval** parameter located under the [pdmq] stanza determines the periodic update interval of the configuration information used by the Tivoli Policy Director for MQSeries server (daemon). The configuration information includes the user mapping, extended attributes, such as recipients and error handling queue, and user information. The parameter **pdmq-cache-interval** is set to a default value of 3600 seconds (one hour) as follows:

```
pdmq-cache-interval = 3600.
```

You may want to modify this value to provide configure updates on a different interval. Setting **pdmq-cache-interval** to 0 or to a negative value (<0) forces an

update every time the attribute or user and group information is accessed. Using a **pdmq-cache-interval** of 0 or less might result in a negative performance impact.

Configuration Changes

In order for any configuration changes made to the configuration file pdmqazn.conf to take effect, you must stop, and then start the Tivoli Policy Director for MQSeries server (daemon) as shown:

```
pdmqd -stop
pdmqd -start
```

Chapter 4. Migration to Tivoli Policy Director for MQSeries, Version 3.8.1

This chapter describes the migration of Tivoli Policy Director for MQSeries, Version 3.8, to the 3.8.1 product version on four platforms: Solaris, AIX, Windows NT, and Windows 2000. This chapter consists of the following sections:

1. "Migrating Tivoli Policy Director for MQSeries on Solaris Platforms"
2. "Migrating Tivoli Policy Director for MQSeries on AIX Platforms" on page 30
3. "Migrating Tivoli Policy Director for MQSeries on Windows Platforms" on page 31

Migrating Tivoli Policy Director for MQSeries on Solaris Platforms

Perform the following steps to migrate Tivoli Policy Director for MQSeries on a Solaris Platform:

1. Stop the MQSeries server.
2. Become root user.
3. Run the **pdmqucfg** command to disable Tivoli Policy Director for MQSeries MQI library interception:

   ```
   pdmqucfg -disable
   ```

4. Run the following commands to remove Tivoli Policy Director for MQSeries, Version 3.8, and the older version of Certificate and SSL Base runtime (gsk5bas):

   ```
   cd/opt
   pkgrm PDMQrte
   pkgrm gsk5bas
   ```

   The following directories remain on your system; do not delete them.

   - /opt/pdmq/etc — This directory contains necessary configuration files.
   - /opt/pdmq/.configure — The current version of Tivoli Policy Director for MQSeries stores its internal state in this directory.
   - /var/pdmq/log — This directory contains log files.
   - /var/pdmq/audit — This directory contains the audit file.
   - /var/pdmq/db — This directory contains the authorization database maintained by the daemon.
   - /var/pdmq/keytab — This directory contains the files pdmqazn.kdb and pdmqazn.sth.

5. Follow the instructions in "Installing Tivoli Policy Director for MQSeries on Solaris" on page 9 to install the current version. When the installation is complete, go to the following step.
6. Run the **pdmqcfg** command to enable Tivoli Policy Director for MQ Series MQI library interception:

   ```
   pdmqcfg -enable
   ```

7. Start the MQSeries server.
8. After you have successfully completed the installation, test the new version of Tivoli Policy Director for MQSeries by sending and receiving messages using the MQPUT and MQGET functions.

9. If you want to enable hardware acceleration support, see "Enabling Hardware Acceleration" on page 159 in Appendix D.

Migrating Tivoli Policy Director for MQSeries on AIX Platforms

Perform the following steps to migrate Tivoli Policy Director for MQSeries on an AIX platform:

1. Stop the MQSeries server.
2. Become root user.
3. Run the **pdmqucfg** command to disable Tivoli Policy Director for MQSeries MQI library intereception:

   ```
   pdmqucfg -disable
   ```

4. Change the current directory to /opt by running the following command:

   ```
   cd /opt
   ```

5. Start the System Management Interface Tool (SMIT). The **System Management** menu is displayed.
6. Select **Software Installation and Maintenance**.
7. Select **Software Maintenance and Utilities**.
8. Select **Remove Installed Software**.
9. Click **List** to display the list of software that can be removed.
10. Select **PDMQ.rte** and **gskkm.rte** and click **OK**.
11. Type `no` in the **PREVIEW only?** text field and click **OK**.
12. Click **OK** in the dialog box that displays the following message:

    ```
    !Continuing may delete information you may want
    to keep. This is your last chance to stop before continuing.
    ```

13. When the uninstallation is complete, click **Done**. The following directories remain on your system and should not be deleted:
    - /opt/pdmq/etc —This directory contains necessary configuration files.
    - /opt/pdmq/.configure — The current version of Tivoli Policy Director for MQSeries stores its internal state in this directory.
    - /var/pdmq/log — This directory contains log files.
    - /var/pdmq/audit — This directory contains the audit file.
    - /var/pdmq/db — This directory contains the authorization database maintained by the daemon.
    - /var/pdmq/keytab — This directory contains the files pdmqazn.kdb and pdmqazn.sth.
14. Follow the instructions in "Installing Tivoli Policy Director for MQSeries on AIX" on page 9 to install the current version of Tivoli Policy Director for MQSeries (**PDMQ.rte**), AIX Certificate, and SSL Base Runtime ACME Toolkit (**gskkm.rte**). When you complete the installation process, go to the following step.
15. Run the **pdmqcfg** command to enable Tivoli Policy Director for MQSeries MQI library interception:

    ```
    pdmqcfg -enable
    ```

16. Start the MQSeries server.
17. After you have successfully completed the installation, test the new version of Tivoli Policy Director for MQSeries by sending and receiving messages using the MQPUT and MQGET functions.

18. If you want to enable hardware acceleration support, see "Enabling Hardware Acceleration" on page 159 in Appendix D.

Migrating Tivoli Policy Director for MQSeries on Windows Platforms

Perform the following steps to migrate Tivoli Policy Director for MQSeries on a Windows platform.

Note: These migration steps use *pdmq_install_path* to refer to the directory where Tivoli Policy Director for MQseries, Version 3.8 or 3.8.1, is installed. By default *pdmq_install_path* is C:\Program Files\Tivoli\Policy Director for MQSeries.

1. Log on as Administrator or as a user with administrator privileges.
2. Stop the MQSeries server.
3. Use the Configuration Wizard to Disable MQI library interception.
4. Change the directory to *pdmq_install_path*.
5. Use **Add/Remove Programs** in the **Control Panel** to uninstall Tivoli Policy Director for MQSeries, Version 3.8.

   The following directories remain on your system and should not be deleted:
   - *pdmq_install_path*\etc — This directory contains necessary configuration files.
   - *pdmq_install_path*\.configure — Tivoli Policy Director for MQSeries stores its internal state in this directory.
   - *pdmq_install_path*\log — This directory contains log files.
   - *pdmq_install_path*\audit — This directory contains the audit file.
   - *pdmq_install_path*\db — This directory contains the authorization database maintained by the daemon.
   - *pdmq_install_path*\keytab — This directory contains the files pdmqazn.kdb and pdmqazn.sth.
6. Follow the instructions in "Installing Tivoli Policy Director for MQSeries on Windows" on page 11 to install the current version.

   Note: By default, Tivoli Policy Director for MQSeries installs files in the C:\Program Files\Tivoli\Policy Director for MQSeries folder.
7. Stop the MQSeries server.
8. Use the Configuration Wizard to Enable MQI library interception.
9. Start the MQSeries server.
10. Test the new version of Tivoli Policy Director for MQSeries by sending and receiving messages using the MQPUT and MQGET functions.
11. If you want to enable hardware acceleration support, see "Enabling Hardware Acceleration" on page 159 in Appendix D.

Chapter 5. Administering Tivoli Policy Director for MQSeries

This chapter describes the details of setting up Tivoli Policy Director for MQSeries, Version 3.8.1, to run in a typical MQSeries environment.

The following tasks are described in this chapter:

Defining and Attaching Policy and Access Control List Templates

The name space configured in Tivoli Policy Director represents the protected queues, which are resources created by MQSeries. You can define policy templates and attach them to these queues. Tivoli Policy Director uses a protected object policy (POP) to specify the quality of protection required on messages flowing through these queues. The POP also specifies the audit level. In addition, Tivoli Policy Director uses access control list (ACL) permission bits to specify who can put messages onto a queue and who can get messages from a queue.

You can use Tivoli Policy Director to define and apply authorization policy templates and create ACL templates. See the *Tivoli SecureWay Policy Director Base Administration Guide* for further information on defining policy and ACL templates.

Specifying Authorization for Tivoli Policy Director for MQSeries Operations

Tivoli Policy Director for MQSeries checks the ACL on a queue when the application attempts to open the queue (using MQOPEN) after connecting to the queue manager. The mode in which the application opens the queue, using either MQPUT or MQGET, determines which permission bit is required. If you do not have the required permissions, then an error is sent back to the API caller, and the MQOPEN request is not passed to the queue manager.

Tivoli Policy Director for MQSeries uses the Tivoli Policy Director action groups to specify the following permission bits on MQSeries queues:

- **E** ("Enqueue") – Represents authority to place messages into a given queue object; it authorizes an entity to call the MQPUT API on the queue.
- **D** ("Dequeue") – Represents authority to retrieve messages from a given queue object; it authorizes an entity to call the MQGET API on the queue.

These permissions are specific to Tivoli Policy Director for MQSeries. They are prefaced by [PDMQ] in the Tivoli Policy Director Web portal manager and in the **pdadmin** command. When you define ACLs for Tivoli Policy Director for MQSeries, you must also set the Traverse (T) bit. ACLs can be attached to queues, queue managers, or the /PDMQ/Queue object.

Note: To perform the following tasks using the Tivoli Policy Director Web portal manager, use the instructions in Appendix A, "Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager" on page 81.

Run the following command to log on to Tivoli Policy Director:

```
pdadmin -a admin_ID -p password
```

Here is an example of a **pdadmin** command that creates an ACL:

```
pdadmin>acl create pdmqacl
```

The following is an example of how to modify an ACL.

Note: You must have already set the T bit using the following command:

```
pdadmin>acl modify pdmqacl set group pdmqusers T[PDMQ]DE
```

In the above example, pdmqusers group has been added to include both Enqueue and Dequeue permissions.

The following is an example of how an ACL may look after modification:

```
pdadmin> acl show pdmqacl
 ACL Name: pdmqacl
Description: PDMQ Test Access Control List
Entries:
User admin TcmdbsvaBl
User pdmquser1 T[PDMQ]ED
User enqueue1 T[PDMQ]E
User dequeue1 T[PDMQ]D
Group pdmqusers T[PDMQ]ED
Group enqusers T[PDMQ]E
Group dequsers T[PDMQ]D
```

In order for an ACL to be effective, you must attach the ACL to an object. The following is an example of how to attach an ACL to a queue in the Tivoli Policy Director protected object space:

```
pdadmin>acl attach /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY pdmqacl
pdadmin>object show /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
Name : /PDMQ Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
Description :
Type : (Non-existent Object) : 10
Is Policy Attachable : yes
ACL : pdmqacl
```

ACL pdmqacl has been attached to the queue /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY.

Specifying the Tivoli Policy Director for MQSeries Protected Object Policy

Quality of protection (QOP) defines how messages are cryptographically protected. For further information, see the *Tivoli SecureWay Policy Director Base Administration Guide*. You can send messages with any of the following levels of protection. The protected object policy (POP) defines these QOP attributes:

- **none** – Indicates that the messages using this queue do not use cryptographic protection. Using this quality of protection does not, in any way, circumvent authorization enforcement on queue operations. The attribute `none` is appropriate for native MQSeries queues. Tivoli Policy Director for MQSeries applies no cryptographic protection to the data.
- **integrity** – Indicates that all messages using this queue have integrity–level protection. Tivoli Policy Director for MQSeries signs the data using the private key of the sending application user. Integrity protection helps prevent the data from being modified without detection.

- **privacy** – Indicates that all messages using this queue have privacy–level protection. Tivoli Policy Director for MQSeries encrypts the data in the public keys of the recipients and signs the data using the private key of the sending application user. This level of protection helps ensure that only authorized recipients can decrypt and read the data. Also, this level of protection confirms the identity of the sender.

In addition, POP specifies the audit level at `all`, `none`, `permit`, `deny`, `admin`, or `error`. For details about these audit level settings, see Chapter 9, "Auditing Tivoli Policy Director for MQSeries" on page 69.

If you do not want any cryptographic protection, specify the QOP attribute `none` in the POP. If a queue in the Protected Object Space does not have a POP specified, all messages sent to that queue have integrity–level protection by default.

For POP entries that specify the QOP of **privacy**, you must also list queue recipients as extended attributes of the queue. This information is needed so Tivoli Policy Director for MQSeries can find the proper encryption keys for the recipients. If the Tivoli Policy Director for MQSeries configuration data does not have the correct recipient names listed, intended recipients are not able to read the message (because they are not able to decrypt it), in spite of having the right permissions to read messages off the queue.

In order to send a message and have it received successfully, both of the following have to be true:

- The sender is authorized to put the message on the queue and is able to correctly protect the message (as specified by the quality of protection in the POP).
- The recipient is authorized to receive the message and correctly validate (and optionally decrypt) the message.

Note: To perform the following tasks using the Tivoli Policy Director Web portal manager, use the instructions in Appendix A, "Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager" on page 81.

Run the following command to log on to Tivoli Policy Director:

```
pdadmin -a admin_ID -p password
```

Here is an example of how to create a POP:

```
pdadmin>pop create pop-pdmq
pdadmin>pop modify pop-pdmq set qop integrity
pdadmin>pop modify pop-pdmq set audit-level none
pdadmin>pop show pop-pdmq
Protected object policy: pop-pdmq
Description:
Warning: no
Audit level: none
Quality of protection : integrity
Time of day access: sun, mon, tue, wed, thu, fri, sat, :anytime:local
IP Endpoint Authentication Method Policy
     Any Other Network 0
```

Here is an example of how to attach a POP to a queue in the Tivoli Policy Director protected object space:

```
pdadmin>pop attach /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY pop-pdmq
pdadmin>object show /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
Name : /PDMQ Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
Description :
Type : (Non-existent Object) : 10
Is Policy Attachable : yes
Policy : pop-pdmq
```

POP pdmq-pop has been attached to the queue /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY.

Specifying Cryptographic Policy and Other Attributes

The type of cryptographic protection (none, integrity, privacy) applied to messages sent to a particular queue is specified in the quality of protection (QOP) attribute of the protected object policy (POP) associated with the queue. Additionally, Tivoli Policy Director for MQSeries allows you to select the particular algorithm or method used to protect the data. Specifying a particular algorithm may be necessary if the certificates and private keys you generated for use by Tivoli Policy Director for MQSeries require a particular algorithm, of if you are concerned about the relative security of one cryptographic method over another.

Tivoli Policy Director for MQSeries classifies encryption algorithms based on the difficulty of defeating the algorithms using brute force methods. Often, this is related to the size of the keys used. The supported algorithms are classified as follows:

- STRONG – Triple DES (also known as 3DES). Triple DES uses a key with an effective length of 168 bits. This is the default algorithm used by Tivoli Policy Director for MQSeries.
- MEDIUM – DES. DES uses a key with an effective length of 56 bits.
- WEAK – RC2. RC2 is used with a key of an effective length of 40 bits.

Note that the algorithms listed above are symmetric algorithms. Tivoli Policy Director for MQSeries encrypts messages with a symmetric key, and then encrypts the symmetric key with the private keys of the recipients. Tivoli Policy Director for MQSeries supports public key encryption using the RSA encryption algorithm with keys of an effective length of up to 2048 bits.

Algorithms used to provide integrity protection (using digital signatures) are simply specified by name. Tivoli Policy Director for MQSeries supports the following digital signature algorithms: MD2, MD5, or SHA-1 (default).

Configuring Extended Attributes

Tivoli Policy Director for MQSeries entries in the object space can have extended attributes set that specify different functionality for Tivoli Policy Director for MQSeries. The extended attributes include the following:

- **Qname-resolution [local | remote]** – Set whether queue name resolution is local or remote.
- **PKI-enc-strength [default | strong | medium | weak]** – Defines the encryption strength to be used when privacy is specified.
- **PKI-sig-algorithm [default | MD2 | MD5 | SHA1]** – Defines the hash algorithm to be used when integrity is specified.
- **Error-handling-Q** *queue_name* – Specifies the queue name of the Tivoli Policy Director for MQSeries error handling queue.

- **Q-enc-strength [default | strong | medium | weak]** – Strength of encryption used to secure messages put onto this queue using MQPUT when privacy is specified. Overrides global setting of PKI-enc-strength.
- **Q-sig-algorithm [default | MD2 | MD5 | SHA1]** – Strength of signing algorithm to use for messages put onto this queue using MQPUT when integrity is specified. Overrides global setting of PKI-sig-algorithm.
- **Q-recipients** ***Cert DN*** – Specifies a recipient for messages put onto this queue using MQPUT. Multiple recipients are specified by repeating the Tivoli Policy Director command to build a name-multi-value pair. MQPUT with privacy will fail unless all of the specified recipients are found in the current identity's KDB file. The distinguished name format is case-sensitive, and you must use a semicolon (;) as a delimiter, for example, CN=pdmquser;O=PISC;C=GB

Configuring Error Handling Queue

Each queue manager in the object space is required to have an error handling queue, which means that each queue manager must have the attribute Error-handling-Q with the specified queue name as its attribute value. Tivoli Policy Director for MQSeries creates this queue automatically when the **pdmqcfg** command is run. Also, you can use MQSeries commands (such as **runmqsc**) or the MQSeries Explorer to create this queue manually. Tivoli Policy Director for MQSeries does not function properly if this attribute is not set.

Note: See Chapter 8, "Tivoli Policy Director for MQSeries Error Handling" on page 67 for more information about the error handling queue.

The configuration command **pdmqcfg** sets the error handling queue attribute for the user. However, if you need to change the attribute manually, you can use the **pdadmin** command. Perform the following steps:

1. Run the following command to log on to Tivoli Policy Director:

   ```
   pdadmin -a admin_ID -p password
   ```

2. If an Error-handling-Q attribute already exists for the queue manager, you must first delete it:

   ```
   pdadmin>object modify /PDMQ/Queue/queue_manager
   delete attribute Error-handling-Q
   ```

3. Then run the following command to change the attribute:

   ```
   pdadmin>object modify /PDMQ/Queue/queue_manager
   set attribute Error-handling-Q queue_name
   ```

   where:

   queue_manager
   Your actual queue manager name.

   queue_name
   The queue name of the Tivoli Policy Director for MQSeries error handling queue.

   Note: After you change the attribute, you must attach pdmq-erq-def-acl and pdmq-erq-def-pop to the error queue. Also, you must be a member of the pdmq-admin group in Tivoli Policy Director in order to browse or get messages from the error queue.

To set the error handling queue using the Tivoli Policy Director Web Portal Manager, follow the instructions in Appendix A, "Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager" on page 81.

Extended Attributes for Encryption Strength, Algorithm, and Recipients

To perform the tasks described below using the Tivoli Policy Director Web Portal Manager, follow the instructions in Appendix A, "Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager" on page 81.

To send messages using an encryption strength other than the default, you must set an extended attribute. At the queue manager level, you specify the following:

```
PKI-enc-strength = strength
```

Or, for each queue:

```
Q-enc-strength = strength
```

where *strength* is either STRONG, MEDIUM, or WEAK (default is STRONG).

To send messages using a signature algorithm other than the default, you must set an extended attribute. At the queue manager level, you specify the following:

```
PKI-sig-algorithm = algorithm
```

Or, for each queue:

```
Q-sig-algorithm = algorithm
```

where *algorithm* is either MD2, MD5, or SHA1 (default is SHA-1).

To send messages with the privacy (encrypted) level of protection for each queue, you must set the following:

```
Q-recipients = distinguished_name_of_recipient
```

Note: Recipients can be set at the queue level only.

The default values of PKI-sig-algorithm and Q-sig-algorithm and PKI-enc-strength and Q-enc-strength are set by the **pdmqcfg** command. For PKI-sig-algorithm and Q-sig-algorithm , the default value is SHA-1. For PKI-enc-strength and Q-enc-strength, the default value is STRONG.

To send messages with integrity–level protection (signed but not encrypted), you do not need to specify Q-recipients and Q-enc-strength.

The following are examples for setting these parameters using the command utility **pdadmin**:

- Run the following command to log on to Tivoli Policy Director:

  ```
  pdadmin -a admin_ID -p password
  ```

- Run the following command to set the signature algorithm parameters:

  ```
  pdadmin object modify /PDMQ/Queue/queue_manager/queue set
  attribute Q-sig-algorithm [MD2|MD5|SHA1]
  ```

  Note: In each case, replace *queue_manager* and *queue* with the actual queue manager and queue names.

  Here is an example of the Tivoli Policy Director object space after Q-sig-algorithm has been added to a queue:

  ```
  pdadmin>object show/PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
  Name: /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY
  Description:
  ```

```
Type : (Non-existent Object) : 10
Is Policy Attachable : yes
Extended Attributes :
Q-sig-algorithm
MD2
```

Queue /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.INTEGRITY has been assigned a signature algorithm (Q-sig-algorithm) of MD2.

- Run the following commands to set the encryption strength:

  ```
  pdadmin object modify /PDMQ/Queue/queue_manager/queue set
  attribute Q-enc-strength [STRONG|MEDIUM|WEAK]
  ```

- Run the following commands to set the queue recipients parameters:

  ```
  pdadmin object modify /PDMQ/Queue/queue_manager/queue set attribute
  Q-recipients "CN=xxx;O=abc;C=us"
  ```

  where CN=xxx;O=abc;C=us is the distinguished name of recipient1.

  ```
  pdadmin object modify /PDMQ/Queue/queue_manager/queue set attribute
  Q-recipients "CN=yyy;0=abc;C=us"
  ```

  where CN=yyy;0=abc;C=us is the distinguished name of recipient2.

  Note: In each case, replace *queue_manager* and *queue* with the actual queue manager and queue names.

When specifying the distinguished name for the certificates of Q-recipients, be sure that component names such as C, CN, O, and OU are specified in uppercase and that each component is separated from the next by a semicolon (;).

You can add multiple recipients by repeating the **pdadmin** command, which sets the Q-recipients attribute.

Here is an example of the Tivoli Policy Director object space after recipients have been added to a queue:

```
Name: /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.PRIVACY
Description :
Type: (Non-existent Object) : 10
Is Policy Attachable : yes
Extended Attributes :
Q-recipients
CN=entrust pdmquser1; O=testbank; C=US
   CN=pdmquser2 cms; O=testbank; C=US
   CN=pdmquser3tpki; O=testbank; C=US
```

In the above example, three queue recipients have been added to the queue /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.PRIVACY

Here is an example of the Tivoli Policy Director object space after recipients and Q-enc-strength (encryption strength) have been added to a queue:

```
Name : /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.PRIVACY
Description :
Type : (Non-existent Object) : 10
Is Policy Attachable : yes
Extended Attributes :
 Q-recipients
CN=get 3 tpki;O=testbank;C=us
CN=get 6 SS;O=testbank;C=US
CN=get 9 racf;O=testbank;C=US
```

```
CN=get 1 entrust;O=testbank;C=US
CN=get 2 cms;O=testbank;C=US
Q-enc-strength
MEDIUM
```

As shown, queue /PDMQ/Queue/b02laix.queue.manager/IN.B02LAIX.PRIVACY has been assigned five recipients and an encryption strength of MEDIUM.

Performing Access Control List Evaluation and Queue Name Resolution

Tivoli Policy Director for MQSeries evaluates access control lists (ACLs) based on the target queue for any MQPUT, MQOPEN, and MQGET operations. When an application specifies a queue manager and queue name combination on a call to MQOPEN, Tivoli Policy Director for MQSeries might resolve the queue manager and queue name pair to the destination queue manager and queue name pair. The extended attribute Qname-resolution associated with the /PDMQ object in the Tivoli Policy Director protected object space determines whether this resolution is performed. If the Qname-resolution attribute is set to `local`, Tivoli Policy Director for MQSeries uses the given queue manager and queue name pair. If the Qname-resolution attribute is set to `remote`, Tivoli Policy Director for MQSeries resolves the queue manager/queue name pair to the destination queue manager and queue name pair.

Note: Neither the **pdadmin** command nor the Tivoli Policy Director Web portal manager validates the name or value of the extended attributes. Therefore you must enter them correctly.

Administering the Tivoli Policy Director for MQSeries Server

The Tivoli Policy Director for MQSeries Server, also referred to as the **daemon** or **pdmqd**, starts automatically as part of the initial complete configuration of Tivoli Policy Director for MQSeries. Similarly, **pdmqd** is stopped automatically as part of the complete unconfiguration process. This section describes the external interface commands exported by the Tivoli Policy Director for MQSeries Server. These commands allow the administrator to manually start or stop the daemon and to perform other administrative tasks without interrupting the service.

Starting pdmqd

Note: Tivoli Policy Director for MQSeries must be configured before you start the Tivoli Policy Director for MQSeries Server, **pdmqd**.

To start **pdmqd** in the background on Solaris and AIX platforms, run the following command:

```
pdmqd -start
```

To start pdmqd on the Windows NT platform, perform the following steps:

1. Click **Start** → **Settings** → **Control Panel** → **Services**. The Services window is displayed.
2. Select **Policy Director for MQSeries Service**.
3. Click **Start**.

To start pdmqd on the Windows 2000 platform, perform the following steps:

1. Click **Start** → **Settings** → **Control Panel** →**Administrative Tools** → **Services**. The Services window is displayed.

2. Right-click **Policy Director for MQSeries Service**. A pop-up menu is displayed.
3. Click **Start** in the pop-up menu.

Starting pdmqd in Foreground (Debug) Mode

Note: Tivoli Policy Director for MQSeries must be configured before you start **pdmqd**.

To start **the Tivoli Policy Director for MQSeries Server, pdmqd**, in the debug or foreground mode run the following command from the command line:

```
pdmqd -foreground
```

Stopping pdmqd

To stop the Tivoli Policy Director for MQSeries Server, **pdmqd**, on Solaris and AIX platforms, run the following command:

```
pdmqd –stop
```

To stop **pdmqd** on the Windows NT platform, perform the following steps:

1. Click **Start** → **Settings** → **Control Panel** → **Services**. The Services window is displayed.
2. Select **Policy Director for MQSeries Service**.
3. Click **Stop**.

To stop **pdmqd** on Windows 2000 platforms, perform the following steps:

1. Click **Start** → **Settings** → **Control Panel** → **Administrative Tools Services** → **Services** for Windows NT. The Services window is displayed.
2. Right-click **Policy Director for MQSeries Service**. A pop-up menu is displayed.
3. Click **Stop** in the pop-up menu.

Getting Version Information

To get the version information about the Tivoli Policy Director for MQSeries Server, **pdmqd**, run the following command:

```
pdmqd -version
```

Updating Configuration Information

Tivoli Policy Director for MQSeries allows the administrator to update the user credentials and the extended attributes information in the memory configuration structures without stopping and restarting the daemon. This feature enables Tivoli Policy Director for MQSeries to provide uninterrupted service to the applications while the administrator is updating the configuration subsystem. The changes which are updated in the configuration subsystem include the following:

- Changes to the map.conf file
- Changes to extended attributes, such as recipients and error handling queue, which are made using the **pdadmin** command or Tivoli Policy Director Web portal manager
- Changes made to user information or group memberships, which are made using the **pdadmin** command or Tivoli Policy Director Web portal manager

To force an update of the configuration structures after making any of these changes, run the following command:

```
pdmqd -update
```

Note: Changes to the extended attributes and user and group membership information are also periodically updated in the configuration subsystem. The update interval is controlled by the configuration parameter **pdmq-cache-interval** located under [pdmq] stanza in the pdmqazn.conf file. The parameter **pdmq-cache-interval** is set to a default value of 3600 seconds (one hour), as shown:

```
pdmq-cache-interval = 3600
```

Setting **pdmq-cache-interval** to 0 or to a negative value (<0) forces an update every time the attribute or user and group information is accessed. Using a **pdmq-cache-interval** of 0 or less might result in a negative performance impact.

Getting a Configuration Dump

To get a snapshot of the configuration structures for the Tivoli Policy Director for MQSeries daemon, **pdmq**, run the following command:

```
pdmqd -dump
```

Note: When you run this command, **pdmqd** must be running.

Getting the Status of pdmqd

To get the status of the Tivoli Policy Director for MQSeries server, **pdmqd**, run the following command:

```
pdmqd -status
```

Chapter 6. Managing Identities for Tivoli Policy Director for MQSeries

This chapter describes how you can manage your Tivoli Policy Director for MQSeries identities. In the following sections, these tasks are described:

- Defining public key infrastructure (PKI) identities for all Tivoli Policy Director for MQSeries users.
- Providing Tivoli Policy Director user registry mappings for all Tivoli Policy Director for MQSeries users.
- Associating (mapping) operating system users to PKI identities.

These tasks are illustrated in the following diagram.

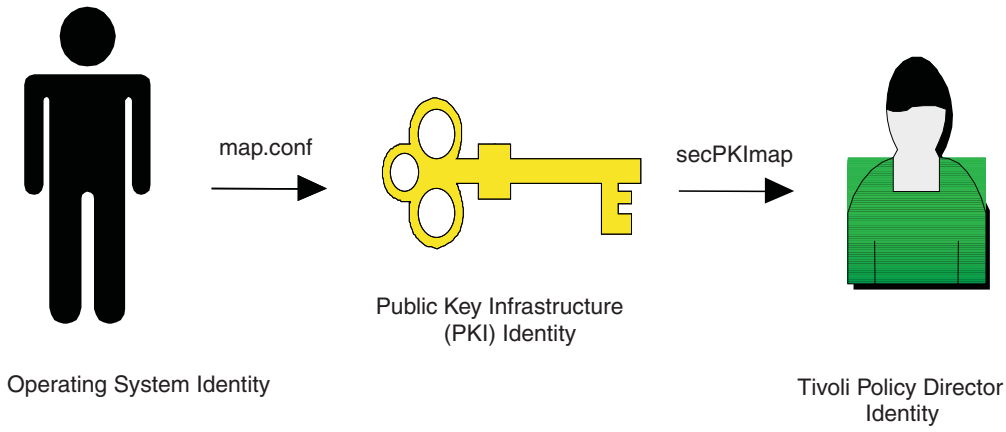

Figure 9. Identity management

Tivoli Policy Director for MQSeries uses two types of identities to represent a user or an application. One type of identity is the PKI identity, which is used for signing and encrypting messages. This identity is represented by the **Distinguished Name** field in a certificate associated with signed and encrypted messages. To authenticate this identity, the user or application must have access to the key database (KDB) file where the certificate and associated private key is stored. Each certificate is represented by a label in the KDB file.

On Windows platforms, the KDB file can be accessed interactively by entering a password. For non-interactive applications (such as server processes) and on UNIX platforms, Tivoli Policy Director for MQSeries provides a mapping file that maps the operating system identity of the user or application to the specific KDB file and certificate label in the KDB file. Furthermore, the mapping file points to a special file called a stash file. The stash file can open the KDB file without the entry of an explicit password.

The second type of identity used by Tivoli Policy Director for MQSeries is a Tivoli Policy Director user identity, which is used to determine if a given user or application has the authority to get or put messages to a queue. This is the identity associated with Tivoli Policy Director access control lists. Tivoli Policy Director for MQSeries ties the certificate identity and the Tivoli Policy Director user identity by

creating a special object in LDAP that links the distinguished name in the certificate to the Tivoli Policy Director user identity.

You can accomplish identity mapping by performing the following steps:

1. Create a PKI identity using Global Security Toolkit (GSKIT) **gsk5ikm** utility. For more information, see "Defining Public Key Infrastructure Identities".
2. Create a Tivoli Policy Director identity using the **pdadmin** command or the Tivoli Policy Director Web Portal Manager. For more information about creating a user, see the *Tivoli SecureWay Policy Director Base Administration Guide*.
3. Map PKI identities to Tivoli Policy Director users. For more information, see "Mapping Public Key Infrastructure Identities to Tivoli Policy Director Users" on page 55.
4. Map operating system users to PKI identities. For more information, see"Mapping Operating System Users to Public Key Infrastructure Identities" on page 60.

Defining Public Key Infrastructure Identities

This section describes key database concepts and provides instructions for performing Global Security Toolkit (GSKIT) GSK5ikm operations.

Key Database Concepts

Tivoli Policy Director for MQSeries uses GSKIT key database (KDB) files for storing certificates and private key information. A typical KDB file will contain a single Personal Certificate, including the private key, a number of Signer Certificates, which identify trusted certificate authorities, and certificates (for data encryption) for other Tivoli Policy Director for MQSeries users.

GSKIT provides a Java-based graphical tool for creating and manipulating KDB files. This is available on all Policy Director for MQSeries platforms. It is started by using the command **gsk5ikm**. The GSKIT tool displays the KDB file into three sections: Personal Certificate Requests, Personal Certificates, and Signers Certificates. Every certificate in the KDB file is identified by a label.

Personal Certificate Requests

When a certificate request is generated, the private key for the request is stored in the *Personal Certificate Requests* section of the KDB file. Keys in this section cannot be used but serve as a list of outstanding requests. The request itself is saved out to a file, which can then be sent to a certificate authority. The certificate authority then builds a certificate from the information in the request and sign it. It is then returned as a certificate file. When the certificate file is loaded into the KDB, the entry in the Personal Certificates Requests section is moved into the *Personal Certificates* section and is ready to use.

Personal Certificates

The Personal Certificates section contains certificates for which the private key is also held. These can be certificates that were previously created as requests (as above), or they may also be certificate/private key data imported from another application. It is also possible to create self-signed certificates, which are useful for testing.

Signer Certificates

The *Signer Certificates* section of the KDB file holds certificates for which the private key is not held. These are either certificate authority certificates or

certificates for other PKI entities. A certificate authority certificate is marked by having a flag set to say it is a "Trusted Root."

Key Database Issues

Certificates used by Tivoli Policy Director for MQSeries must have the key usage fields in the certificate set appropriately by the certificate authority. For certificates used for integrity, the key usage field must be set to **nonRepudiation** and **digitalSignature**. For data privacy, the key usage bits must also include the **dataEncipherment** setting.

With a self-signed certificate, the Subject of the certificate is the same as the Issuer. Therefore, you can only load a certificate into the KDB file if it is either a self-signed certificate or if the certificate of the Issuer is already present in the Signer section of the KDB file. GSKIT handles the key usage fields appropriately for self-signed certificates.

When a KDB file is first created, it is preloaded with a number of well-known certificate authority certificates. If you do not need these certificate authorities for your Tivoli Policy Director for MQSeries installation, then they should be removed. This can be done by selecting each certificate and clicking on **Delete**.

Performing Global Security Toolkit Operations

To use Tivoli Policy Director for MQSeries, you must create a public-private key pair and a certificate. Also, you must have the certificate of the certificate authority that issues the user or application certificate imported into the client key database file and marked as a trusted root.

Creating a Key Database File

The initial step in preparing the PKI identities is to create a key database file.

1. Type `gsk5ikm` to start the Java utility. For Solaris platforms, the path is /opt/ibm/gsk5/bin; for AIX platforms, the path is /usr/opt/ibm/gskkm/bin; and for Windows platforms, the path is \program files\ibm\gsk5\bin. The gsk5ikm utility needs a working Java installation. To verify your installation, run **java –version**.
2. Click **Key Database File**.
3. Click **New**. Click **Open** if the key database already exists.
4. Specify the key database file name and location, and click **OK**. See "Creating a Key Database File".

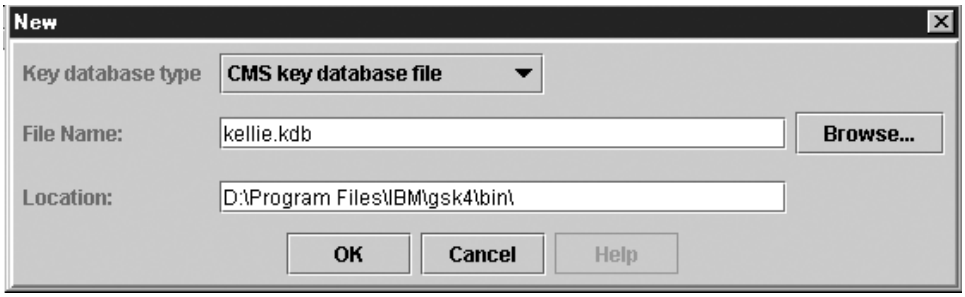

Figure 10. Create a key database file

5. Specify a password for the KDB file and confirm the password.

   This password is used to protect the private key information in the KDB file and will be required in order to open the KDB file for editing.

6. Check the box labeled **Stash the password to a file?**
7. Click **OK**. See Figure 11.

Note: For the UNIX platforms, Tivoli Policy Director for MQSeries only uses stash files to open the key database file, instead of prompting the user for a password. On Windows NT, the administrator configures whether a user will be prompted for the password or if the application will use a stash file. See "Mapping by Using Interactive Login (Windows Only)" on page 61.

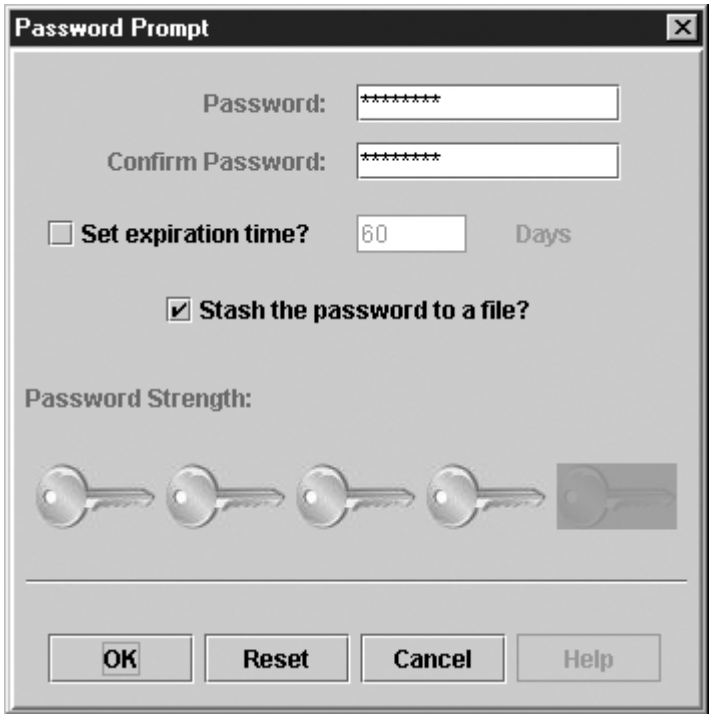

Figure 11. Enable stash file support

User key database files must have the appropriate read and execute permissions in order for users to execute the MQSeries applications. You specify the location of the key database files in the map.conf file.

You should change the key database password frequently. If you specify an expiration date, you should keep track of when you need to change the password. If the password expires before you change it, the key database is not usable until the password is changed.

Attention: Possession of a stash file and the associated key database is sufficient to impersonate the application associated with the public-private key pair and certificate stored in the key database. Ensure that these files are accessible only to the owner.

Creating a Self-signed Certificate

A self-signed certificate is one where the signer of the certificate is also the subject (the one identified by the certificate). This means that the public key in the certificate entity corresponds to the private key that signed it.

A self-signed certificate can be used for anything that a normal certificate can be used for, since the public and private keys have the same properties. The difference is in how the validity of the certificate is verified. Self-signed certificates are good for testing because they can be generated locally without the need for a certificate authority.

If you want to use self-signed certificates, you can create them with the GSKIT gsk5ikm utility. To create self-signed certificates, perform the following steps from the ″Personal Certificate″ view:

1. Under **Key Database Content**, select **Personal Certificates**.
2. Click the **New Self Signed...** button, which displays a certificate request dialog.
3. In the **Key Label** field, specify a label that will identify the new certificate within the KDB. This value must be unique within the KDB.
4. Specify other details that will be included in the certificate.
5. Click **OK**.

   The GSKIT gsk5ikm utility generates a key pair. One key is designated as the private key and the other is the public key. The information provided on the dialog and the public key is formatted and signed using the private key to produce a certificate. The certificate and private key are then stored in the Personal Certificates part of the KDB identified by the label.

   When a certificate is added to the Personal Certificates part of the KDB, and it is not the first to be added, a dialog is shown asking if the new certificate would be marked as the default certificate. This simply means that the certificate is the one offered first when the KDB is opened. It has no other effect.

Receiving a Certificate Authority Certificate

Before any non-root certificate can be loaded into the KDB, the signer of the certificate must be loaded into the KDB as a signer certificate.

1. Under Key Database Content, click **Signer Certificates**. Click **Add**. See Figure 12 on page 48.

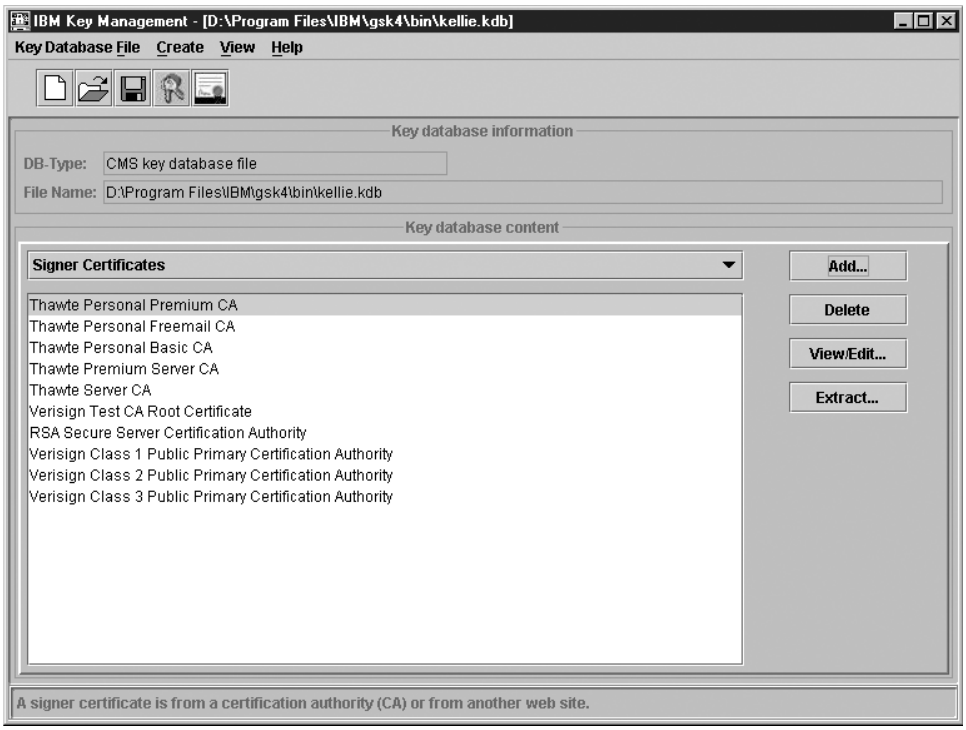

Figure 12. Add a certificate authority certificate

2. Select the data type of the certificate authority certificate you want to add. The data type must be one of the following:
   - Binary Distinguished Encoding Rules (DER) data
   - Base 64-encoded ASCII data

   Your PKI administrator can tell you which data type to use.
3. Specify the file name and location of the certificate authority certificate. See Figure 13.

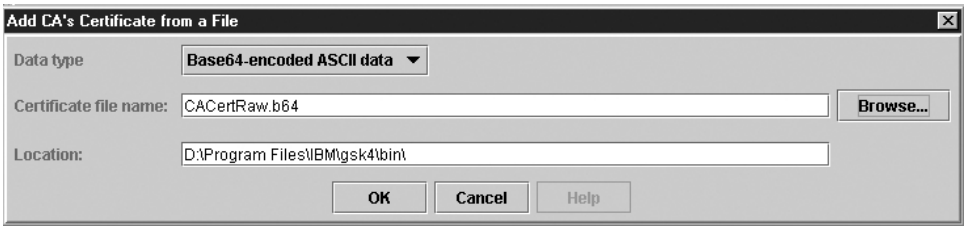

Figure 13. Specify certificate authority certificate type

4. Click **OK**.
5. As prompted, enter a label for the certificate. See Figure 14 on page 49. The label identifies the certificate authority certificate in the key database file.

Figure 14. Enter certificate authority certificate label

6. Select **OK**. You now see the certificate you requested in the list of Signer Certificates. See Figure 15.

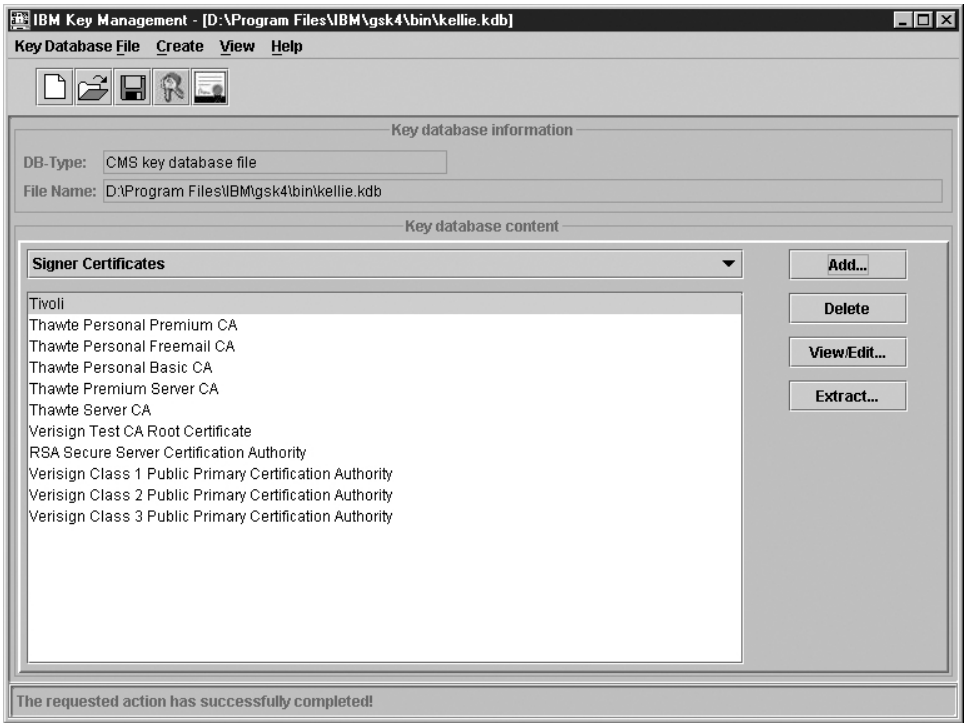

Figure 15. Display requested certificate in list

7. To mark the certificate as a trusted root (a certificate authority certificate that you agree to trust), click **View/Edit**. Select the check box labeled **Set the certificate as a trusted root**. For example, see Figure 16 on page 50.

Figure 16. Set certificate as trusted root

Creating a Certificate Request

The GSKIT gsk5ikm utility tool can be used to create a certificate request.

1. Under Key Database Content, click **Personal Certificate Requests**. See Figure 17 on page 51.

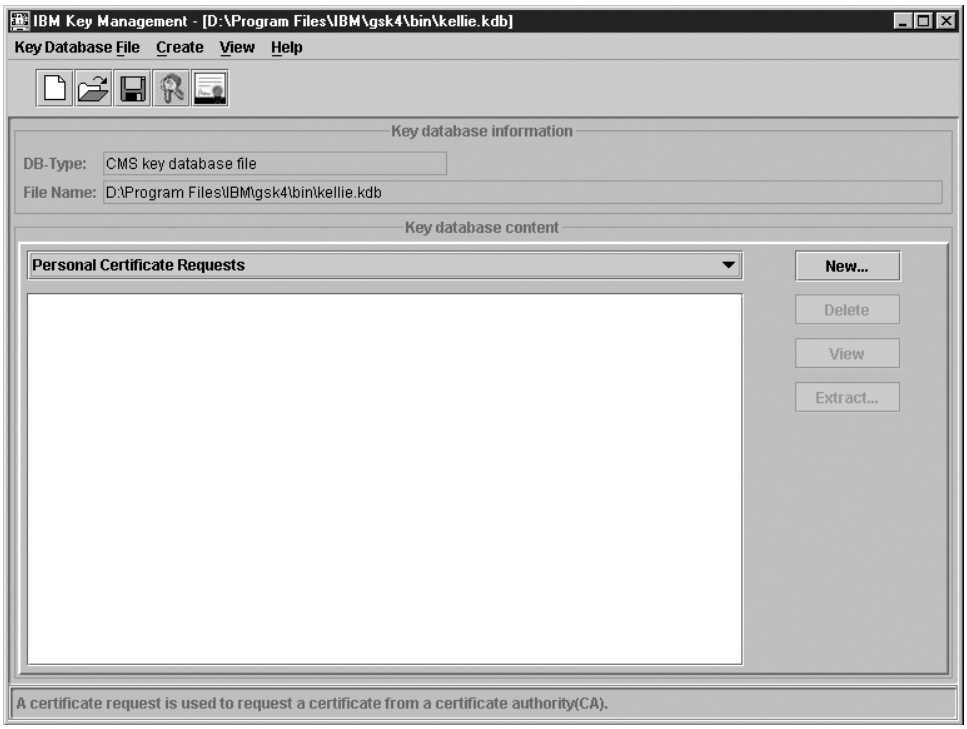

Figure 17. Request personal certificate

2. Click **New**. See Figure 18.

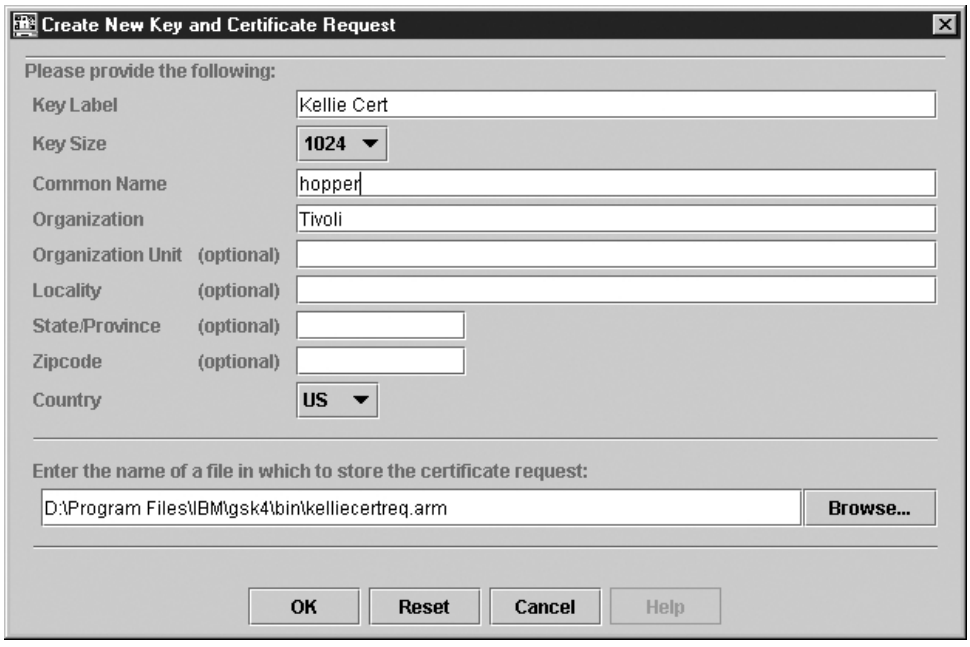

Figure 18. Enter data for certificate request

3. Supply a user-assigned label for the key pair. The label identifies the key pair and certificate in the key database file.
4. Enter the appropriate information for your environment. This information specifies the details that will be included in the certificate.
5. Click **OK**.

6. A message identifying the name and location of the certificate request file is displayed. See Figure 19. Click **OK**.

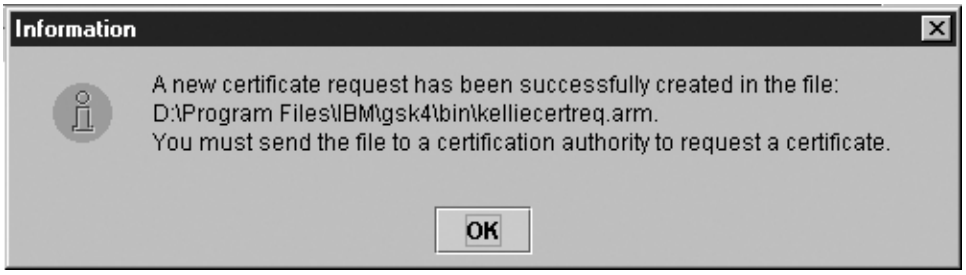

Figure 19. Location of certificate request

The certificate request is now displayed in the list of outstanding requests. See Figure 20.

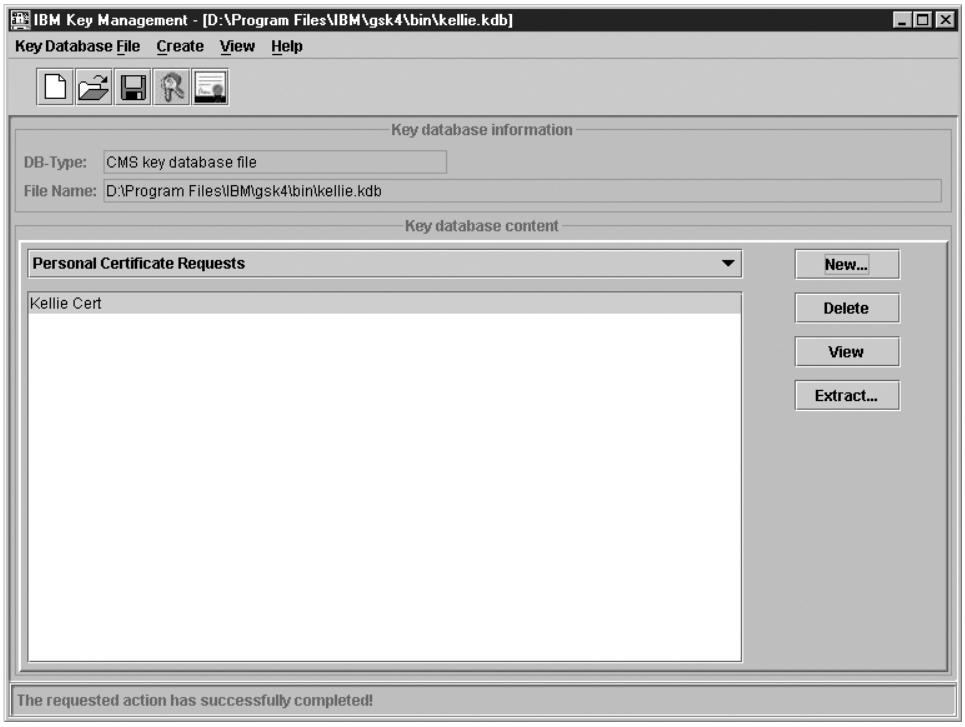

Figure 20. Certificate request displayed in list

Send the certificate request to the certificate authority. The manner in which you send your certificate request to the certificate authority and have it approved depends upon how your PKI administrator sets up the PKI for your organization.

Receiving Your Certificate from the Certificate Authority

1. Select **Personal Certificates** under **Key database content**. See Figure 21 on page 53.

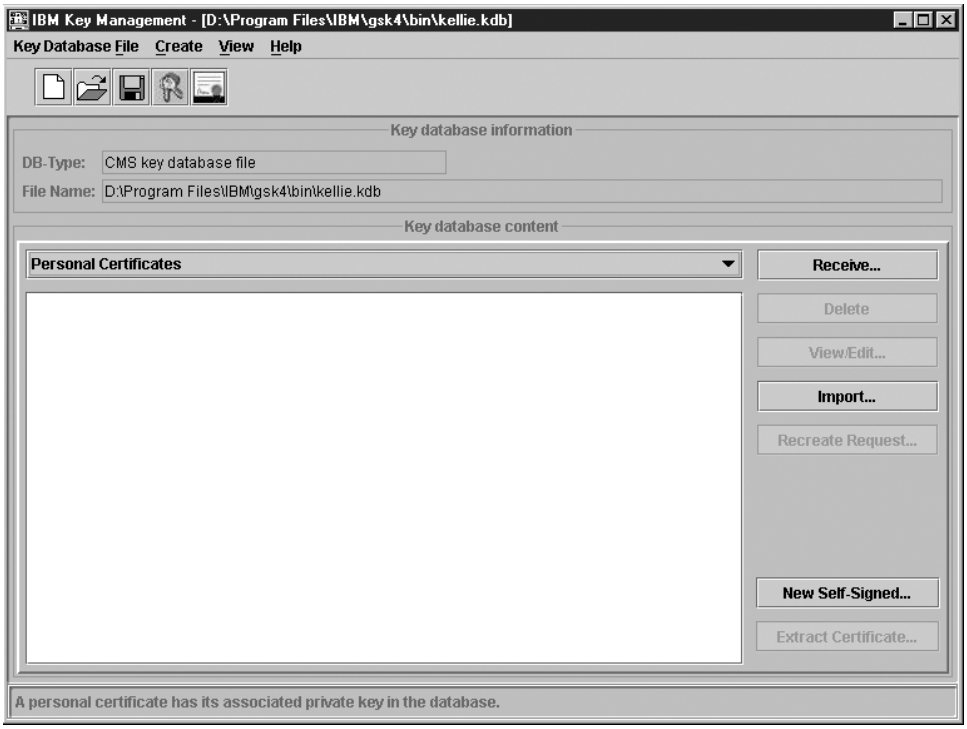

Figure 21. Receive personal certificate

2. Click **Receive**.
3. Enter the file name and location of the certificate you received from the certificate authority and click **OK**. See Figure 22.

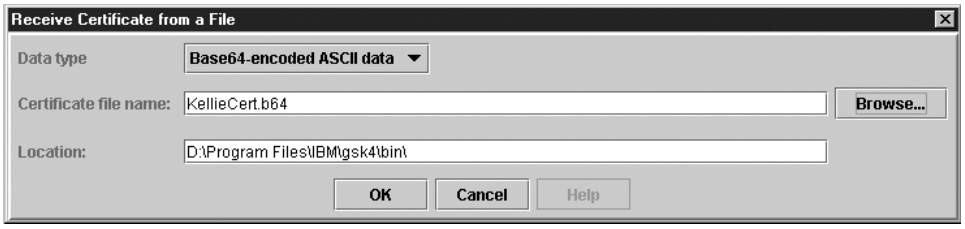

Figure 22. Enter certificate name and location

The certificate is now displayed in your list of certificates. See Figure 23 on page 54.

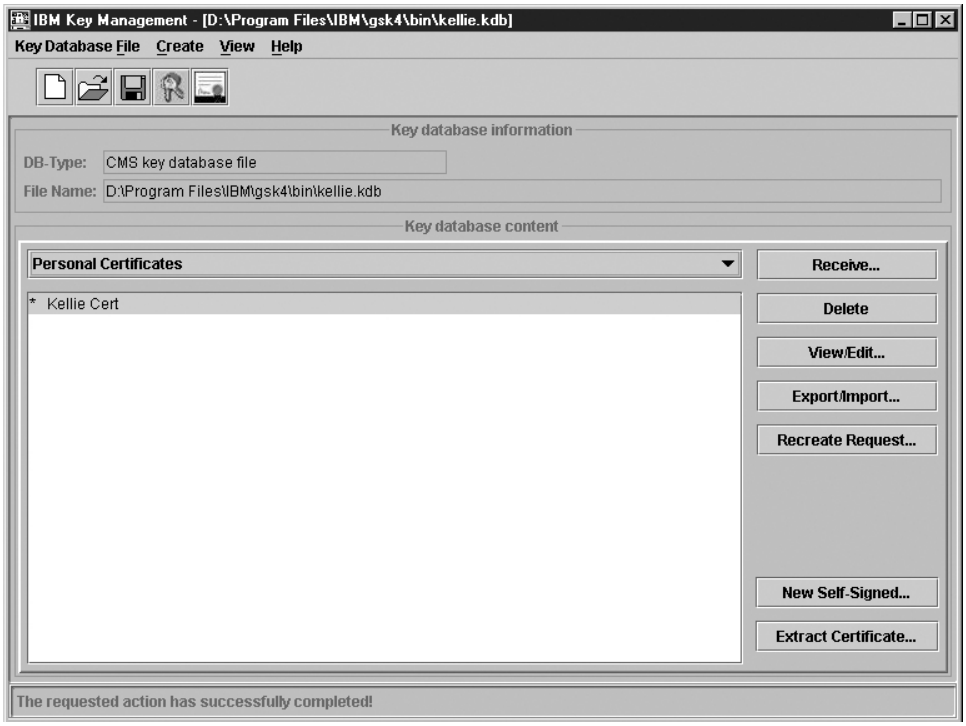

Figure 23. Certificate request completed

Importing Public-Private Key Pairs and Certificates

The gsk5ikm utility also has the ability to import public-private key pairs and certificates that were previously generated using the Public–Key Cryptography Standards (PKCS) #12 format.

1. Under **Key database content**, select **Personal Certificates**. See Figure 22.
2. Click **Export/Import**.
3. Choose **Action Type - Import Key**.
4. Enter the file name and location of the PKCS12 file and click **OK**.
5. Enter the password for the PKCS12 file and click **OK**. The certificate is now displayed in your list of certificates. See Figure 22.

Adding Other Signer Certificates and Certificates for Encryption

In order to validate signatures, Tivoli Policy Director for MQSeries requires that the signer certificates be imported into the key database. In addition, if an application needs to send an encrypted message, Tivoli Policy Director for MQSeries requires that the recipient's certificate be imported into the key database. The steps to do this are almost exactly the same with one critical difference. For certificates for encryption, do not select the check box labeled **Set the certificate as a trusted root** after you import the application or end-user certificate.

The following is the procedure to add other signer certificates and certificates for encryption:

1. Click **Signer Certificates**. See Figure 11.
2. To load a new certificate, click the **Add...** button, which displays the Add CA's Certificate from a File dialog. See Figure 12.
3. Specify the name and location of the file containing the certificate and click **OK**.

4. Specify a unique label that will identify the certificate within the KDB file. See Figure 13.
5. If the certificate is a certification for encryption and not a signer, select the certificate. Then select **View/Edit**. Ensure that the **Set the certificate as a trusted root** box is not checked. See Figure 16 on page 50

Mapping Public Key Infrastructure Identities to Tivoli Policy Director Users

Tivoli Policy Director for MQSeries requires that Tivoli Policy Director be configured to store user and group information into an LDAP directory. Figure 24 shows a typical installation of Tivoli Policy Director user and group information in an LDAP directory.

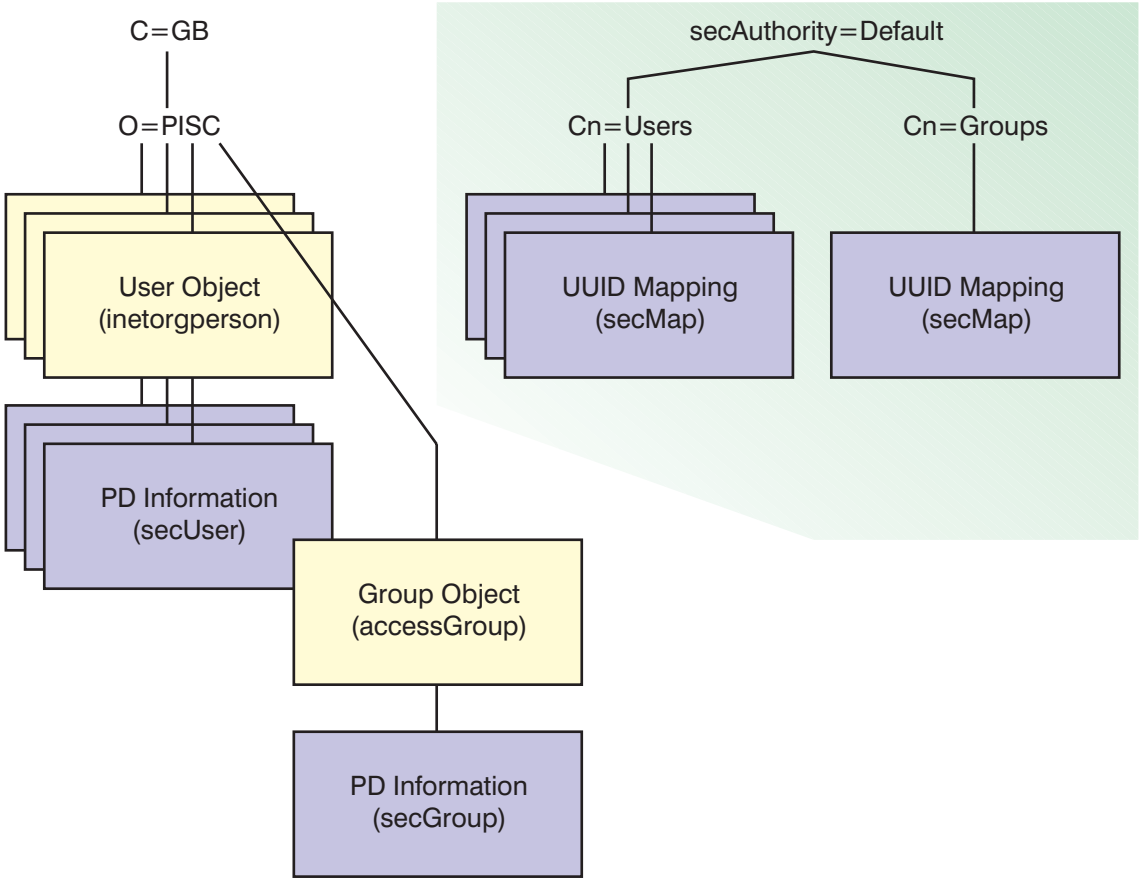

Figure 24. A typical Tivoli Policy Director tree in LDAP

The **secMap** objects store links between a Tivoli Policy Director user entry in LDAP and the representation of the Tivoli Policy Director user in the Tivoli Policy Director authorization database. The mapping model is illustrated in Figure 25 on page 56.

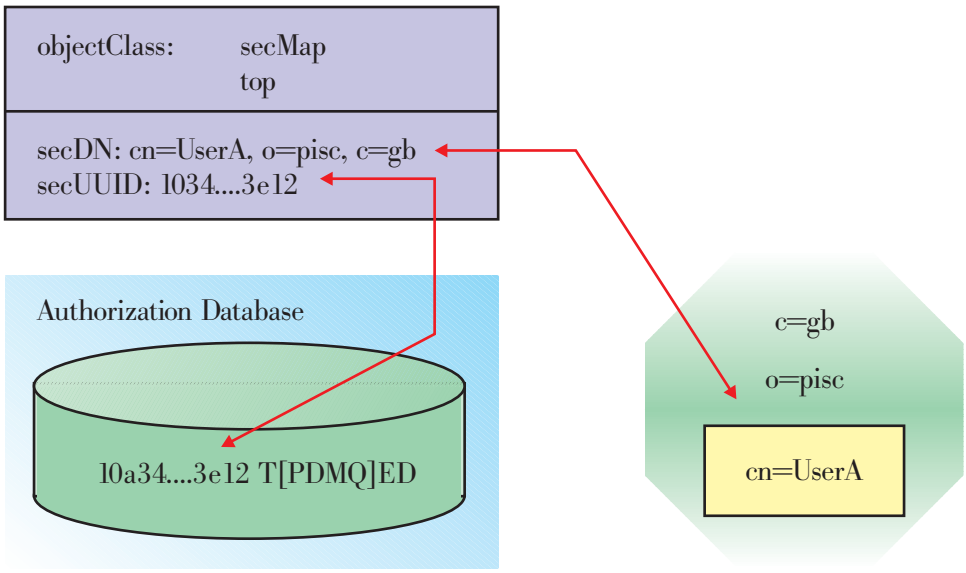

Figure 25. Using the secMap object to tie Tivoli Policy Director user entries to authorization database information

The authorization database represents users and groups as Universal Unique Identifiers (UUIDs). When you use either Tivoli Policy Director Web Portal Manager or the **pdadmin** command to create a Tivoli Policy Director user, the following are also created:

- The user entry in LDAP
- The appropriate **secMap** object
- A UUID which references the user entry in LDAP

Tivoli Policy Director for MQSeries extends the **secMap** object to link a user's certificate to the user's Tivoli Policy Director user entry by adding an auxiliary object to the **secMap** object. This auxiliary object has an LDAP object class of **secPKIMap**. The relationship between **secMap** and **secPKI map** is shown in Figure 26 on page 57.

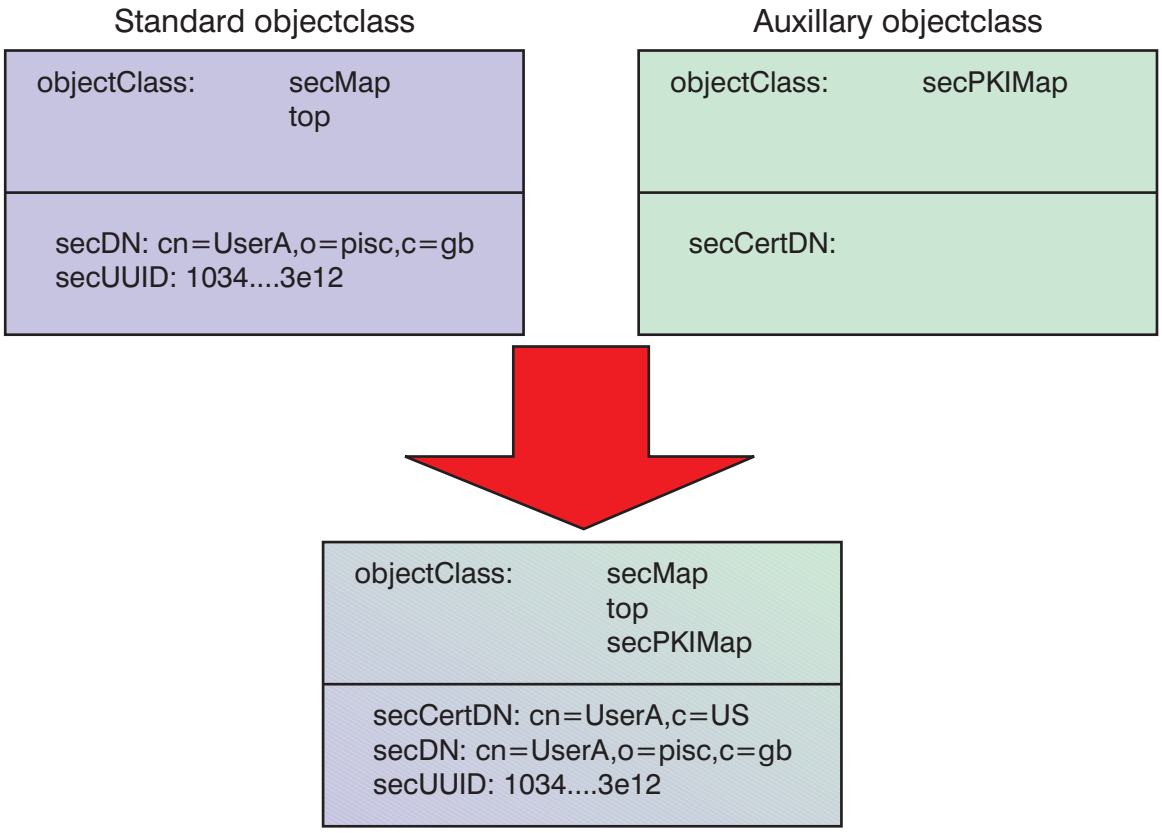

Figure 26. Extending the secMap object with secPKIMap

The **secCertDN** attribute in the **secPKIMap** object contains the distinguished name of the user's PKI certificate. When Tivoli Policy Director for MQSeries uses a PKI identity in a certificate from the KDB file, it searches the **secMap** objects to find the one whose **secCertDN** attribute matches the distinguished name of the certificate.

Creating the secPKIMap Object Class in LDAP

Before you configure Tivoli Policy Director for MQSeries, Version 3.8.1, you must create the **secPKIMap** object class in LDAP by using the **ldapmodify** command, which passes the pdmq.ldif file as input. The pdmq.ldif file is located in the /doc directory of the Tivoli Policy Director for MQSeries CD. Since Chapter 2 of this manual also describes how to create the **secPKIMap** object class, you may have already performed this task.

To create the **secPKIMap** object class using the **ldapmodify** command, perform the following steps:

1. From the MS-DOS command prompt on your machine, run the following command:

   ```
   ldapmodify -h hostname -p 389 -D admin_ID -w password -f
   \pdmq_install_path\etc\pdmq.ldif
   ```

   where:

   hostname
   The host name of the machine being configured.

   389 The default LDAP port number.

admin_ID
The LDAP administrator ID specified earlier.

password
The LDAP administrator password specified earlier.

Adding secPKIMap Objects to Existing secMap Objects

Adding secPKIMap objects to existing secMap objects allows the mapping of the Tivoli Policy Director user to the user's PKI certificate using the distinguished name in the certificate. This task can be performed using the Directory Management Tool (DMT)

Perform the following steps to accomplish this task:

1. Open the tree of objects.
2. Highlight the particular **secMap** object to be updated.
3. Click **Add auxiliary class**.

   Note: The **Add auxiliary class** button may be hidden to the far right of the screen.

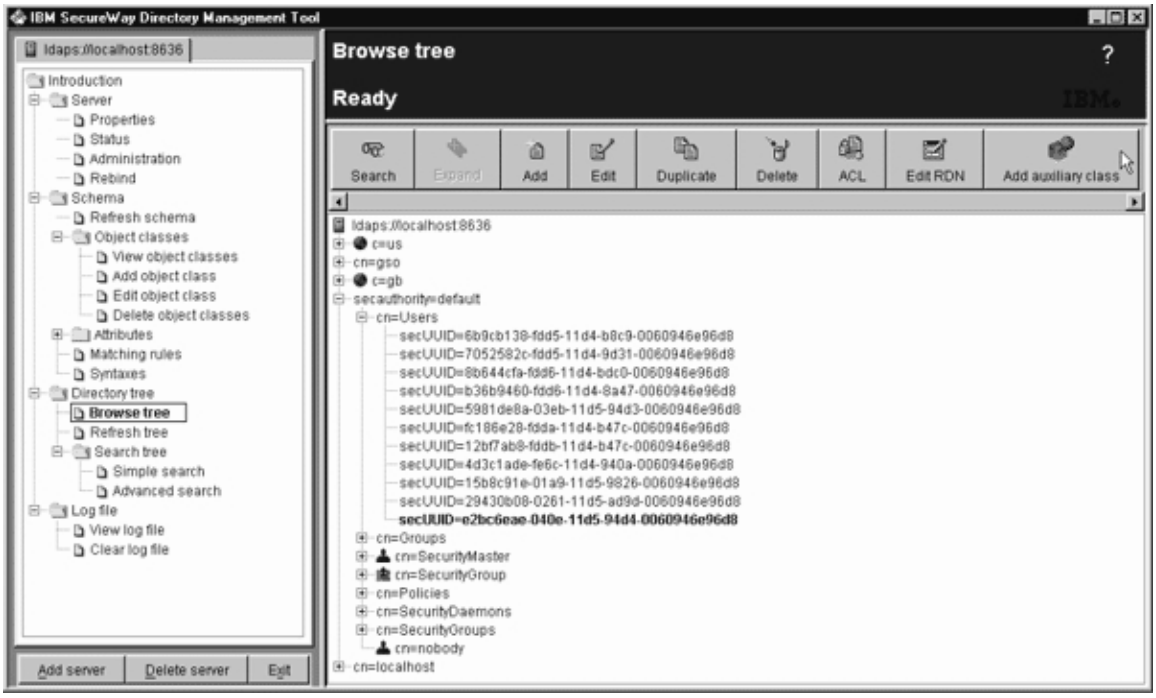

Figure 27. Browsing the tree of existing secMap objects

Note: You can also find a particular object using the Search capability of DMT, using the distinguished name of the Tivoli Policy Director user as the search key for the **secDN** attribute in the tree of **secMap** objects.

4. Highlight **secPKIMap** in the list of available auxiliary classes.

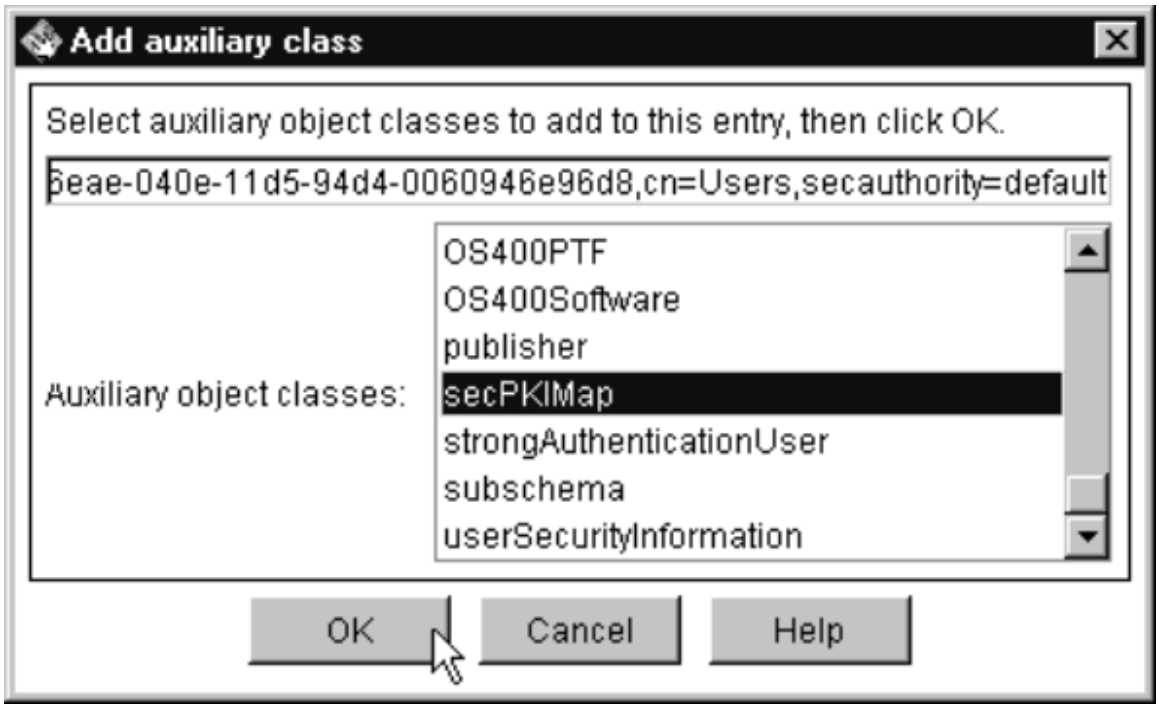

Figure 28. Attaching a secPKIMap object to an existing secMap object

5. Edit the **secMap** object entry, entering the **secCertDN** object that matches the distinguished name in the certificate to be mapped to this Tivoli Policy Director user. The entry is case-sensitive and should match exactly.

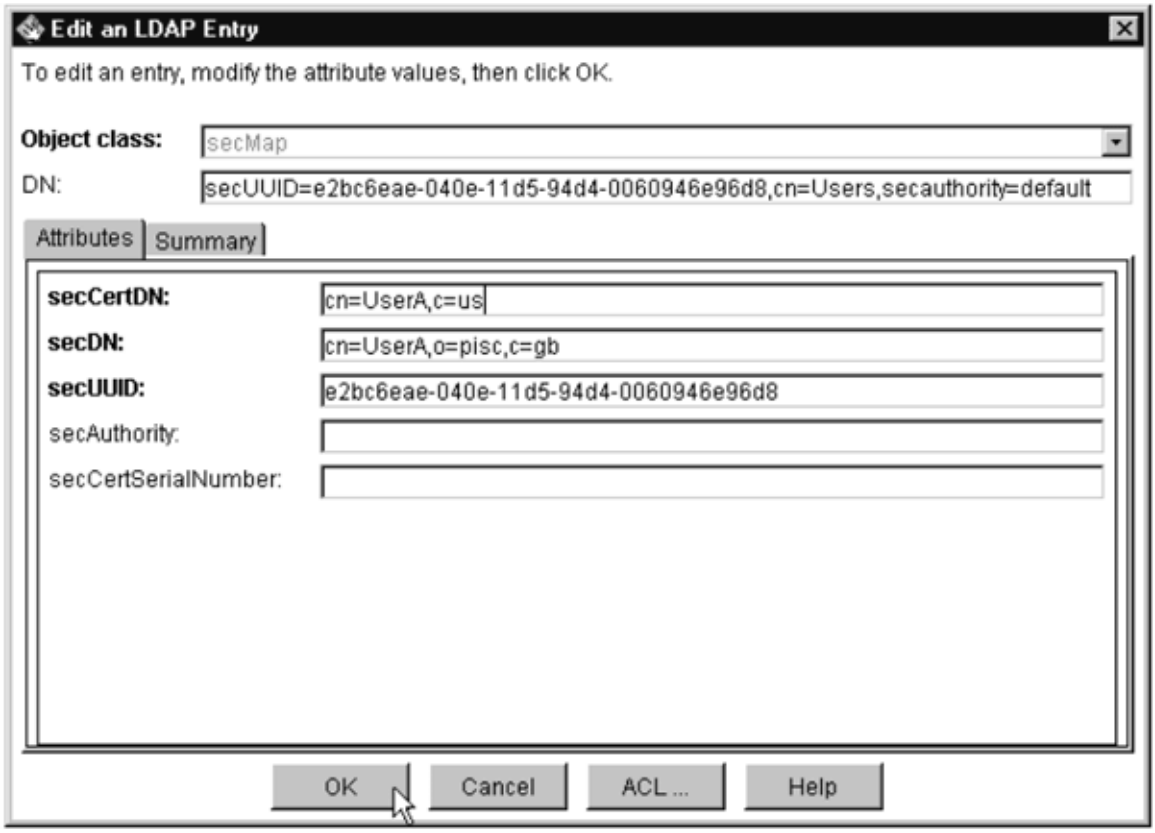

Figure 29. Updating the secPKIMap object with the certificate data for this Tivoli Policy Director user

Mapping Operating System Users to Public Key Infrastructure Identities

Identities using MQ applications must be mapped to a PKI user so that Tivoli Policy Director for MQSeries can access your user's PKI identity, which is needed to send signed messages and to decrypt incoming messages.

The following sections describe two methods of mapping your login to a PKI user.

Mapping Operating System Users to Public Key Infrastructure Users

There are two ways to create the necessary mapping:

- Process login: This method is used for non-interactive login when all the mapping information is pre-defined in a configuration file in which each operating system user is mapped to a PKI user. This method is the default and is supported on all operating systems. See "Mapping by Using the Process Method (Supported on All Platforms)" on page 61.
- Interactive login: This method can only be used on a Windowsplatform when you access an MQ application running on the local machine. See "Mapping by Using Interactive Login (Windows Only)" on page 61.

Mapping by Using the Process Method (Supported on All Platforms)

The process login enables Tivoli Policy Director for MQSeries to access your operating system identity for the MQSeries application and to define this identity as the PKI identity. As a result, the PKI identity represents the identity for MQSeries application, and thus all users running the application. If the MQ application is running locally, then the PKI identity is specific to the local machine and thus to the local user.

The */pdmq_install_path*/etc/map.conf file contains a list of operating system identities and corresponding information mapping them to a PKI identity, each of which is in the following format:

```
operating_system_identity path_name label
```

where:

operating_system_identity
: Specifies the name of your operating system.

path_name
: Specifies the path of your key database file without the KDB extension.

label
: Specifies the label in the KDB file.

The following are examples of entries in the map.conf file:

```
Administrator,c:\pdmq,pdmquser1
MQappOSID,C:\pdmq2,MQapplabel
pdmquser,c:\Program Files\Tivoli\PDMQ\etc\pdmqkey,PDMQ Self Signed Cert
```

When you try to access a queue that is protected by Tivoli Policy Director for MQSeries, the following sequence of actions is performed:

1. Tivoli Policy Director for MQSeries searches the list of operating system identities in */pdmq_install_path*/etc/map.conf until it finds an entry that matches the current operating system user (for example, Administrator). If it does not find a match, the MQ request fails and an error message is displayed.

   Note: You can use any text editor to edit the map.conf file. After you have changed the map.conf file, you must run **pdmqd –update** so that the Tivoli Policy Director for MQSeries server (daemon) can pick up the changes.
2. If a match is found (the specified KDB file exists) and there is a stash file for the KDB file in the same directory, the specified KDB file is opened. Otherwise, the MQ request fails and an error message is displayed.
3. The personal certificate associated with the KDB file, specified by the label in the KDB file within the map.conf file, is then returned as the certificate for Tivoli Policy Director for MQSeries to use.

Mapping by Using Interactive Login (Windows Only)

This method is used for interactive login where you are prompted for mapping information. Once you create a mapping, it is used for all MQSeries processes that are running under your particular user identity. You are also prompted for the period of time that Tivoli Policy Director for MQSeries will use the selected mapping. Once this period expires, you have to refresh the mapping again.

By default, the interactive login is off on windows. To turn the interactive login feature on, type:

```
C:\>setlogin on
```

To turn the interactive login off and thus turn the process method on, type:

```
C:\>setlogin off
```

To create an active mapping when the interactive login is turned on, you must complete the following steps:

1. Perform one of the following to display the Public Key Infrastructure Login window:
   - Click **Start → Programs → Policy Director for MQSeries → Policy Director for MQSeries Login**.
   - Run **pdmqlogin** from the command line.
   - Try to access a queue that is protected by Tivoli Policy Director for MQSeries when there is no existing active mapping.

   Note: If you try to access a queue that is protected by Tivoli Policy Director for MQSeries and you have an active mapping already, your request succeeds and the PKI login is not displayed.
2. Enter the file path and file name of your KDB file in the **Key Database File** field.
3. Enter the password in the **Password** field.

   Note: This password is the one that you specified when you created the KDB file.
4. Click **Login** to display the Personal Certificates Available window.
5. The default certificate from the KDB file is displayed in the Personal Certificates Available window. To select another certificate, display the drop-down list and click the certificate you need to use.
6. In the **Time to live (in seconds)** field, specify the number of seconds you need the login context to be valid. For example, if you specify `360`, then the mapping will be valid for 6 minutes.
7. Click **OK** to return to the Public Key Infrastructure Login window.
8. Click **Exit**.

If you need to discontinue the Tivoli PKI login current mapping before that time, perform the following steps:

1. Open the Public Key Infrastructure Login window using one of the following:
   - Click **Start → Programs → Policy Director for MQSeries → Policy Director MQSeries Login.**
   - Run `pdmqlogin` from the command line.

   2. Click **Logout**.

Chapter 7. MQSeries Considerations

This chapter describes how you can use Tivoli Policy Director for MQSeries to provide enhanced authorization and data protection to messages.

Managing Authorization Interaction between MQSeries in Tivoli Policy Director for MQSeries

The authorization applied by Tivoli Policy Director for MQSeries does not replace the authorization applied by MQSeries itself, but rather enhances it. Access to protected queues is first checked by Policy Director for MQSeries, and if granted, access is then checked by the Queue Manager. Thus, MQSeries administrators assign permissions to queues through both Tivoli Policy Director for MQSeries and MQSeries to ensure that a protected queue can be accessed. Assigning permissions locally to MQSeries queues by using the SETMQAUT command is described in the *MQSeries System Administration* manual, which is available at the following Web site:

http://www-4.ibm.com/software/ts/mqseries/library/manualsa/manuals/crosslatest.html

Resetting Tivoli Policy Director for MQSeries after Applying Service Fixes to MQSeries

To provide authorization and data protection to messages, Tivoli Policy Director for MQSeries relocates the MQI shared library supplied with MQSeries and replaces it with a Tivoli Policy Director for MQSeries–specific version of the MQI shared library. To accomplish this task, run the **pdmqcfg** command. Before applying a service fix to MQSeries on either a Solaris or AIX platform, run the following command:

```
pdmqucfg -disable
```

After the service fix has been applied to MQSeries, for Solaris and AIX platforms, run the following command to restore the Tivoli Policy Director for MQSeries versions of the MQI shared library:

```
pdmqcfg -enable
```

On Windows platforms, perform the following steps before applying a service fix to MQSeries:

1. Click **Start** → **Programs** → **Policy Director for MQseries** → **pdmqcfg**.
2. Click **Disable Interceptor**.

After the service fix has been applied to MQSeries on Windows platforms, perform the following steps:

1. Click **Start** → **Programs** → **Policy Director for MQseries** → **pdmqcfg**.
2. Click **Enable Interceptor**.

Resetting Maximum Message Size in Tivoli Policy Director for MQSeries

One of the attributes an MQSeries administrator can set on a queue (or queue manager) is the MaxMsgLength. This attribute indicates the longest physical length a message can be in order for it to be put in a queue. Because Tivoli Policy Director for MQSeries increases the size of messages by adding a secure encapsulation to them, Tivoli Policy Director for MQSeries encapsulated messages might exceed the MaxMsgLength limit. If a message does exceed this limit, MQSeries rejects it with the return code MQRC_MSG_TOO_BIG_FOR_Q or MQRC_MSG_TOO_BIG_FOR_Q_MGR. To address this problem, MQSeries administrators must increase the value of MaxMsgLength.

Using Tivoli Policy Director for MQSeries with MQSeries Clusters

Tivoli Policy Director for MQSeries fully supports MQSeries clusters. Tivoli Policy Director for MQSeries does not distinguish between cluster queues and non-cluster queues. Therefore, each queue in the cluster must be added to the protected object space using **pdmqcfg** to enable Tivoli Policy Director for MQSeries to secure the queue. Thus, each cluster queue is represented in the protected object space by an entry underneath the queue manager where the queue is hosted.

For example, suppose that there are two queue managers (AUSTIN and DALLAS) that are in a cluster called TEXAS. If the AUSTIN queue manager hosts the cluster queue WATERLOO and the DALLAS queue manager hosts the cluster queue REDRIVER, the protected object space would have the following entries:

```
/PDMQ/Queue/AUSTIN/WATERLOO
/PDMQ/Queue/DALLAS/REDRIVER
```

MQSeries clusters support a special type of queue called workload balance queues, where a queue with the same name resides on two or more different queue managers. When an application puts a message to the queue, MQSeries determines which queue manager will be sent the message. Because Tivoli Policy Director for MQSeries cannot determine which queue manager will receive the message, the same ACL and POP must be applied to all the instances of a workload balance queue. Furthermore, the same set of Q-recipients must be specified if the quality of protection in the POP is set to privacy.

To extend the previous example, if the queue managers AUSTIN and DALLAS hosted the cluster queue CACTUS, the protected object space would have the following entries:

```
/PDMQ/Queue/AUSTIN/WATERLOO
/PDMQ/Queue/AUSTIN/CACTUS
/PDMQ/Queue/DALLAS/REDRIVER
/PDMQ/Queue/DALLAS/CACTUS
```

The requirements to have matching security policy and Q-recipients also applies to cluster alias queues and their base queues, and to cluster remote queues and their base queues.

For more information about MQSeries clustering, refer to *MQSeries Queue Manager Clusters*.

Managing Unsupported MQSeries Configurations

Tivoli Policy Director for MQSeries does not support the following MQSeries options:

- Channel data conversion
- Application message segmentation
- Use of Message Reference Header messages (MQRMH)
- Use of non-threadsafe MQI libraries
- MQSeries clients

Chapter 8. Tivoli Policy Director for MQSeries Error Handling

This chapter describes how Tivoli Policy Director for MQSeries uses error handling queues to process invalid messages.

Error Handling Queue

Tivoli Policy Director for MQSeries routes any invalid messages received to an error handling queue. Invalid messages are those with one or more of the following conditions:

- The sender did not have the authority to write to the queue.
- The sender's certificate was invalid.
- A policy mismatch occurred, because the sender either used integrity instead of privacy or the wrong algorithm.
- A message was sent without Tivoli Policy Director for MQSeries encapsulation from a regular MQSeries machine.

Note: Tivoli Policy Director for MQSeries will not issue any warnings if an application sends a message to a queue using an expired certificate. However, the recipient will be unable to retrieve the message from the queue, and the message will be placed on the error queue.

You must either use **pdmqcfg** to create the error queue or create it manually using MQSeries. If you do not define a custom error queue, the error messaging system will not function. Once your error queue is set up, all invalid messages are delivered to it.

When Tivoli Policy Director for MQSeries sends a message to the error handling queue, the following message is returned:

```
MQGET ended with reason code 2109 (MQRC_SUPPRESSED_BY_EXIT).
```

dlqutil Utility

The **dlqutil** utility is an interactive program that enables MQSeries administrators to examine messages that Tivoli Policy Director for MQSeries places in the error handling queue. Only users who belong to the pdmq-admin group can browse the messages in this queue. An administrator who belongs to this group can either browse all the messages or search for a particular message based on the following criteria:

- Reason code
- Queue manager and queue name to whom the message was sent
- Application name, application put date, application put time

The command syntax for **dqutil** is as follows:

```
dlqutil error_queue_name [queue_manager]
```

where:

error_queue_name
: The name of the error handling queue that was specified during configuration.

queue_manager
The name of the queue manager or the default queue manager name if not specified.

When you select a message in the Tivoli Policy Director for MQSeries error handling queue, you can display the message, including the reason why it was put on the queue, or copy the message to a file.

Chapter 9. Auditing Tivoli Policy Director for MQSeries

Auditing is the process of recording the system activities or events that affect the secure operation of Tivoli Policy Director for MQSeries. Any attempt to access a protected resource or perform any operation on a protected resource that can change the state of the system constitutes an audit event and is recorded in the audit trail. An audit trail is a collection of audit records.

If auditing is specified, Tivoli Policy Director for MQSeries writes audit records in a file in the audit directory under the installed Tivoli Policy Director for MQSeries directory structure. Tivoli Policy Director for MQSeries generates a single audit file for the system.

Configuring Auditing

Auditing in Tivoli Policy Director for MQSeries is configured through the following parameters which are located under the [aznapi-configuration] stanza in the pdmqazn.conf file:

logaudit

To enable auditing, set the logaudit parameter to yes as shown:

```
[aznapi-configuration]
logaudit=yes
```

Note: The default setting for logaudit is yes.

auditlog

Use the auditlog parameter to specify the full path of the audit trail file. By default, these paths are set as follows:

On Solaris and AIX platforms:

```
auditlog = /var/pdmq/audit/audit.log
```

On Windows platforms:

```
auditlog = pdmq_install_path\audit\audit.log
```

logsize

The administrator can specify a maximum size (in bytes) for the audit trail file by setting the logsize parameter to the desired size. Upon reaching this maximum size, also called the rollover size, the auditlog file is time-stamped and renamed audit.log.*timestamp*. All subsequent records are written to a new audit.log file.

The various possible logsize values are defined as follows:

If logsize is set to a negative value (< 0), then a new audit trail file is created with each invocation of the auditing process and every 24 hours from that instance.

If logsize is set to equal zero (= 0), then no rollovers are performed and the audit trail file grows indefinitely. If an audit trail file already exists, new records are appended to it.

If logsize is set to a positive value (> 0), then a rollover is performed when an audit trail file reaches the configured rollover value. If an audit trail file already exists at startup, new records are added to it.

The default rollover size is as follows:

```
logsize = 2000000
```

The number of rolled over files is limited only by the disk space available in the file system.

logflush
Audit trail files are written to the buffered data streams and are flushed to the disk at an interval configured by the `logflush` parameter. Changing the flush interval may be required when the audit trails are being examined in real time. By default, audit trail files are flushed every 20 seconds, as shown:

```
logflush = 20
```

Specifying a negative value for the `logflush` parameter forces each audit record to be flushed to the disk as it is written.

For more information about the audit configuration parameters, refer to the *Tivoli SecureWay Policy Director Base Administration Guide*.

Specifying Audit Level for Tivoli Policy Director for MQSeries

Tivoli Policy Director administrators can use the **pdadmin** command to specify what level of auditing is to be recorded for access to a resource. The audit level is specified in the protected object policy (POP) and can be set to any of the following.

Note: See "Specifying the Tivoli Policy Director for MQSeries Protected Object Policy" on page 34 for more information about the POP.

all Records all auditable events for Policy Director for MQSeries.

none Does not record any Tivoli Policy Director for MQSeries events. Auditing function is turned off.

permit Records only successful accesses to Tivoli Policy Director for MQSeries–protected resources.

deny Records only denied requests for access to Tivoli Policy Director for MQSeries–protected resources.

admin Records OPEN, CLOSE, PUT, and GET operations on protected MQSeries queues.

error Records any unsuccessful GET operations which result in messages being sent to the error handling queue.

Here are some examples of how to specify audit levels:

```
pdadmin>pop modify pop_name set audit-level all
pdadmin>pop modify pop_name set audit-level none
pdadmin>pop modify pop_name set audit-level permit
```

You can also specify multiple audit levels. For example:

```
pdadmin>pop modify pop_name set audit-level permit,deny,error
```

Understanding the Audit Trail File Format

The audit trail file contains a set of XML style entries. The XML file is in ASCII and can be read directly or passed to other external parsing engines for further analysis. For more information about the XML format and the suggested document type definition (DTD), refer to the ″Audit Trail File Format″ section in the *Tivoli SecureWay Policy Director Base Administration Guide*.

Audit Record Description

This section describes the audit records generated by Tivoli Policy Director for MQSeries and the operations that are audited.

The following is a sample audit record for Tivoli Policy Director for MQSeries generated for a GET operation:

```
<event rev="1.1">
<date>2001-10-04-17:40:56.187+00:00I-----</date>
<outcome status="0">0</outcome>
<originator blade="pdmq"><component rev="1.1">pdmq</component>
<action>0</action> <!-- Event id -->
<location>bart</location> <!-- Hostname -->
</originator>
<accessor name="administrator">
<principal auth="IV_LDAP_V3.0">pdmquser3</principal>
</accessor>
<target resource="5">
<object>/PDMQ/Queue/bart1.qm/TEST.QUEUE</object></target>
<data>
</data>
<data tag="action">MQGET</data>
<data tag="operation">D</data>
<data tag="result">MQGET call successful</data>
<data tag="prot-operation">sign only</data>
<data tag="sign-algorithm">SHA1</data>
<data tag="encode-algorithm">DEFAULT</data>
<data tag="originator">/C=US/O=Tivoli/CN=tivolipki pdmq cert</data>
<data tag="MsgId">414d512062617274312e716d202020203bbc996000002013</data>
<data tag="MsgFormat">MQSTR   </data><!-- Message Format -->
<data tag="ProcessId">29079</data><!-- Process Id -->
<data>
</data>
</event>
```

Each record generated by Tivoli Policy Director for MQSeries consists of a single event. Each event starts with the `event rev` tag and has two distinct sections:

1. Common audit data
2. Event–specific data

Common Audit Data

The first section section of the audit record contains information that is common to all events. Common audit data starts from the event `timestamp` and ends with the object description. In the following example of the common audit data, each entry is described briefly in XML style comments:

```
<event rev="1.1">
<date>2001-10-04-17:40:56.187+00:00I-----</date>
<!-- Event time stamp -->
<outcome status="0">0</outcome>
<!-- outcome of the operation and status;
     0 - success, 1 - failure -->
<originator blade="pdmq"><component rev="1.1">pdmq</component>
<!-- component name and revision number -->
```

```
<action>0</action> <!-- Event id -->
<location>bart</location> <!-- Hostname -->
</originator><!-- Name of user being audited -->
<accessor name="administrator">
<principal auth="IV_LDAP_V3.0">pdmquser3</principal>
<!-- authentication mechanism and principal name -->
</accessor>
<target resource="5">
<!-- resource operation type code;
     0 - Authorization,  3 - Credential, 5 - General -->
<object>/PDMQ/Queue/bart1.qm/TEST.QUEUE</object></target>
<!-- Object being audited -->
```

Event-Specific Data

For event-specific data, tags are added at the end of the common audit data section. More details about event-specific data for various types of events is provided in the next section. The following is an extract from the previous example record for a GET operation:

```
<data>
</data>
<data tag="action">MQGET</data>
<!-- operation: OPEN, PUT, GET, CLOSE  -->
<data tag="operation">D</data>
<data tag="result">MQGET call successful</data>
<data tag="prot-operation">sign only</data>
<data tag="sign-algorithm">SHA1</data>
<data tag="encode-algorithm">DEFAULT</data>
<data tag="originator">/C=US/O=Tivoli/CN=tivolipki pdmq cert</data>
<!-- sender -->
<data tag="MsgId">414d512062617274312e716d202020203bbc996000002013</data>
<!-- Message Id -->
<data tag="MsgFormat">MQSTR   </data><!-- Message Format -->
<data tag="ProcessId">29079</data><!-- Process Id -->
<data>
</data>
```

Auditable Events in Tivoli Policy Director for MQSeries

This section provides a description of all the Tivoli Policy Director for MQSeries auditable events and the specific data it generates for each event in the event-specific data section.

Authorization Check in MQOPEN

Before a protected queue can be opened Tivoli Policy Director for MQSeries performs an authorization check to determine the permissions to access the queue. If the audit level includes `permit`, all authorization checks returning an ″access granted″ decision will be audited. If the audit level includes `deny`, all authorization checks returning an ″access denied″ decision will be audited.

Tivoli Policy Director for MQSeries generates the following event-specific data, along with a brief explanation (in XML style comments) for each entry:

```
<data tag="action">MQOPEN</data>
<!-- MQSeries call being executed -->
<data tag="operation">E</data>
<!-- operation for which access is allowed/denied
     possible values; E – Enqueue, D - Dequeue -->
<data tag="result">access allowed</data>
<!-- access decision
     Possible values; access allowed,
          access denied or authorization check failure -->
<data tag="qop">privacy</data>
```

```
<!-- quality of protection on the queue
     Possible QOP levels; none, integrity, privacy  -->
<data tag="ProcessId">4800</data>
<!-- Process Id of the application being audited -->
```

Actual MQOPEN Operation

If the audit level includes admin, a successful MQOPEN operation is audited. Tivoli Policy Director for MQSeries generates the following event-specific data:

```
<data tag="action">MQOPEN</data>
<!-- MQSeries call being executed -->
<data tag="operation">N</data>
<!-- operation is non-relevant -->
<data tag="result">MQOPEN call successful</data>
<!-- result of the MQOPEN operation  -- >
<data tag="ProcessId">4800</data>
<!-- Process Id of the application being audited -->
```

Actual MQPUT Operation

If the audit level includes admin, a successful MQPUT operation is audited. Tivoli Policy Director for MQSeries generates the following event-specific data:

```
<data tag="action">MQPUT</data>
<!-- MQSeries call being executed -->
<data tag="operation">E</data>
<!-- operation being performed; E - Enqueue -->
<data tag="result">MQPUT call successful</data>
<!-- result of the MQPUT operation  -- >
<data tag="prot-operation">sign and encrypt</data>
<!-- protection operation performed on the message
possible values; 'sign only' for integrity, 'sign and encrypt' for privacy
     not generated for no protection  -->
<data tag="sign-algorithm">DEFAULT</data>
<!-- sign algorithm when QOP is integrity or privacy
     not generated for no protection -->
<data tag="encode-algorithm">DEFAULT</data>
<!-- encode algorithm when QOP is integrity or privacy
     not generated for no protection -->
<data tag="recipients">CN=tivolipki pdmq cert;O=Tivoli;C=US
</data>
<!-- recipients list, generated only when QOP is privacy -->
<data tag="MsgId">414d512062617274312e716d202020203bc6059600004013</data>
<!-- message Id of the message enqueued -->
<data tag="MsgFormat">MQSTR   </data>
<!-- format of the message enqueued -->
<data tag="ProcessId">14522</data>
<!-- Process Id of the application being audited -->
```

Actual MQGET Operation

If the audit level includes admin, a successful MQGET operation is audited. Tivoli Policy Director for MQSeries generates the following event-specific data:

```
<data tag="action">MQGET</data>
<!-- MQSeries call being executed -->
<data tag="operation">D</data>
<!-- operation being performed; D - Dequeue -->
<data tag="result">MQGET call successful</data>
<!-- result of the MQGET operation  -- >
<data tag="prot-operation">sign and encrypt</data>
<!-- protection operation performed on the message
     possible values; 'sign only' for integrity, 'sign and encrypt' for
     privacy
     not generated for no protection  -->
<data tag="sign-algorithm">SHA1</data>
<!-- sign algorithm when quality of protection is integrity or privacy
```

```
     not generated for no protection -->
<data tag="encode-algorithm">STRONG</data>
<!-- encode algorithm when quality of protection is integrity or privacy
     not generated for no protection -->
<data tag="originator">/C=US/O=Tivoli/CN=tivolipki pdmq cert</data>
<!-- sender of the message -->
<data tag="MsgId">414d512062617274312e716d202020203bc6059600004013</data>
<!-- message Id of the message dequeued -->
<data tag="MsgFormat">MQSTR   </data>
<!-- format of the message dequeued -->
<data tag="ProcessId">14543</data>
<!-- Process Id of the application being audited -->
```

Sender's Authorization Check for Received Message

For all messages received from the protected queues Tivoli Policy Director for MQSeries performs an authorization check to determine if the sender was indeed allowed to PUT the message in the queue. This authorization check is not done for queues with quality of protection value of `none`. If the audit level includes `permit` or `deny`, Tivoli Policy Director for MQSeries audits the event and generates the following data:

```
<data tag="action">MQGET</data>
<!-- MQSeries call being executed -->
<data tag="operation">E</data>
<!-- operation for which access was allowed/denied -->
<data tag="result">sender was authorized</data>
<!-- result of the sender's authorization check -->
<data tag="qop">privacy</data>
<!-- quality of protection -->
<data tag="ProcessId">14543</data>
<!-- Process Id of the application being audited -->
```

Actual MQCLOSE Operation

If the audit level includes `admin`, a successful MQCLOSE operation is audited. Tivoli Policy Director for MQSeries generates the following event–specific data:

```
<data tag="action">MQCLOSE</data>
<!-- MQSeries call being executed -->
<data tag="operation">N</data>
<!-- operation is non-relevent -->
<data tag="result">MQCLOSE call successful</data>
<!-- result of MQCLOSE operation -->
<data tag="ProcessId">4800</data>
<!-- Process Id of the application being audited -->
```

Error Condition in an MQGET Operation

When an error occurs during receipt of a message from the protected queue, the message is put in the error handling queue. This event is audited if the audit level includes the `error` value. Tivoli Policy Director for MQSeries generates the following event–specific data:

```
<data tag="action">MQGET</data>
<!-- MQSeries call being executed -->
<data tag="operation">D</data>
<!-- operation being performed; D - Dequeue -->
<data tag="result">MQGET call failed</data>
<!-- result of MQGET operation -->
<data tag="reason">QoP mismatch</data>
<!-- reason for failure of MQGET operation
     see below for a complete list of reason codes -->
<data tag="MsgId">414d512062617274312e716d202020203bc6059600005013</data>
<!-- message Id of the message dequeued -->
<data tag="MsgFormat">MQSTR   </data>
```

```
<!-- format of the message dequeued -->
<data tag="ProcessId">14543</data>
<!-- Process Id of the application being audited -->
```

Table 1 describes some of the possible reasons for errors that occur when messages are received.

Table 1. Possible reasons for error

Message	Explanation
QOP mismatch	Quality of protection mismatch exists.
security error	Sender is not authorized to put message on the queue.
decrypt error	Message cannot be decrypted.
sender mapping error	Public Key Infrastructure user cannot be mapped to the Tivoli Policy Director user.
sender azn check failed	Authorization check for sender of the message failed.
pdmq header error	Tivoli Policy Director for MQSeries message header cannot be accessed.
no IDUP env established	An Independent Data Unit Protection (IDUP) environment is not established for this session.
size mismatch	A message size mismatch exists.
encryption algorithm strength mismatch	Message encryption algorithm is weaker than required.
unknown error	Unknown error exists.

Chapter 10. Serviceability, Debugging and Tracing Messages

Tivoli Policy Director for MQSeries produces serviceability and debugging and tracing messages for service purposes. The first type of message includes event and status information about system activity, either normal or extraordinary. This type of message is called serviceability information. It is useful to the system administrator and the end user for determining system status. The second type of message is used for troubleshooting purposes (normally when the system is not configured properly or malfunctions). These messages are known as debugging and tracing messages.

Serviceability Messages

Tivoli Policy Director for MQSeries categorizes the serviceability messages into four severity levels:

- FATAL: Fatal error has occurred. System cannot continue. For example, `load library failed`.
- ERROR: Error has occurred, but system can still continue.
- WARNING: Recoverable error has occurred.
- NOTICE: Status report or other information about a system event, such as system startup or shutdown.

Tivoli Policy Director for MQSeries logs FATAL, ERROR, and WARNING events for the Tivoli Policy Director for MQSeries server (`pdmqd`) activities in the *pdmq_install_path*`/log/pdmqd.log` file. For each application that calls the Policy Director for MQSeries interceptor, the events with FATAL, ERROR, and WARNING levels of severity are logged in the *pdmq_install_path*/log/*app_name -process_id_number*.log file. The events with the NOTICE level of severity can be logged manually by adding the following entry in the Tivoli Policy Director message routing file:

```
NOTICE:FILE:/var/pdmq/log/notice.log
```

Note: In Tivoli Policy Director, Version 3.7.1, the file resides at *pd_install_path*/var/svc/routing. In Tivoli Policy Director 3.8, the file resides at *pd_install_path*/etc/routing. Refer to the "Logging and Auditing Server Activity" chapter of the *Tivoli SecureWay Policy Director Base Administration Guide*.

The following is an example of a serviceability message:

```
2001-10-15-17:14:35.538+00:00I----- 0x34D8307E PDMQD NOTICE mqd init daemon.cpp 424
0x00000001 DRQDD0126I :Policy Director for MQSeries Server Shutdown succeeded
```

The message includes the following components:

- A time stamp in Greenwich mean time format, which indicates when the event occurred
- The 32–bit internal message ID in hexadecimal format
- The application name
- The severity level
- The component name
- The subcomponent name
- The source file name

- The line number in the source file
- The thread ID
- The IBM message ID
- The message text

An example of an IBM message ID is DRQDD0126I. The IBM message ID contains the following components:

- Designation for the product name (DRQ in the example, for Tivoli Policy Director for MQSeries).
- Subcomponent name (DD in the example).
- Four-digit message number (0126 in the example).
- Severity label (I in the example, for ″Information″)

Note: There are three IBM-specific severity labels: I (Information), W (Warning), and E (Error).

Debugging and Tracing Messages

Tivoli Policy Director for MQSeries collects debugging and tracing messages so that the IBM service team can troubleshoot problems when necessary. Normally these messages are turned off. You must manually add routing information in order to turn them on, and you need to turn them on only when an IBM service representative advises you to do so. Improper use may impair system performance and require the use of more system resources.

There are nine severity levels defined for debugging messages and there are three severity levels for tracing. As the severity level number increases, the amount of detail provided in the messages increases. These are the three severity levels for tracing:

- Application programming interface level
- Component external interface level
- Component internal interface level

The tracing severity levels are mapped to debug severity levels 4, 6, and 8, respectively.

The message routing control is set in the routing file, but the syntax for routing is:

comp:*subcomp.sev_level*:*where:parameter*[;*where*:*parameter*]

where:

comp Component name

subcomp
Subcomponent name

sev_level
Debugging severity level, from 1 to 9

where:parameter
Routing destination type and its parameter. The three valid options for `where` are as follows:

- `FILE`, which routes the message to a specified file and for which `parameter` is the file path of the file
- `STDOUT`, which routes the message to a standard output and for which `parameter` is always `-`

- STDERR, which routes the message to a standard error output and for which parameter is always -

For example, the command mqm:open.9:STDOUT:-;FILE:/var/pdmq/log/mqm.log sets up a message route for all debugging and tracing messages of severity level less than or equal to 9 for subcomponent open of component mqm. It outputs these messages to STDOUT and to the file /var/pdmq/log/mqm.log.

If you want to route messages for all the subcomponents of a component, use the wildcard mqm:*.9, where mqm in this example is the name of the component, instead of the subcomponent name.

There are four components in Tivoli Policy Director for MQSeries:

- mqm: Tivoli Policy Director for MQSeries library interceptor
- mqd: Tivoli Policy Director for MQSeries server
- mqt: Tivoli Policy Director for MQSeries tools
- dlh:Tivoli Policy Director for MQSeries error handling tool

Components mqm and mqd are the two major components. The component mqm has the following subcomponents:

- general: generic errors
- mqinit: initialization events
- mqconn: MQCONN, MQDISC API events
- mqopen: MQOPEN, MQCLOSE API events
- mqput: MQPUT, MQPUT1 API events
- mqget: MQGET API events
- mqconfig: configuration events
- message: messaging events
- utility: other system utility events

For example, if you want to troubleshoot a problem related to a MQGET() API call, set the following route to dump all debug and tracing information to the file /var/mqm/log/mqget.log:

```
mqm:mqget.9:FILE:/var/pdmq/log/mqget.log
```

The administrator must check the file permissions of the log files that contain debug information. If multiple applications using different user IDs attempt to log debugging information, each user must have write permission to the log file. Add the following line to the Tivoli Policy Director routing file to specify that each application must create its own log file for debugging information:

```
mqm:*.9:FILE:/var/pdmq/log/mqm-%ld.log
```

This specification creates a log file for debug information using the application process identifier (PID).

Appendix A. Administering Tivoli Policy Director for MQSeries Using Tivoli Policy Director Web Portal Manager

This appendix provides specific information about how to use the Tivoli Policy Director Web Portal Manager to perform Tivoli Policy Director for MQSeries administrative tasks. For more information about the Tivoli Policy Director Web Portal Manager, refer to the *Tivoli SecureWay Policy Director Web Portal Manager Administration Guide*.

Tivoli Policy Director Web Portal Manager, Version 3.8, is available as part of the Tivoli SecureWay Policy Director, Version 3.8, CD package. You can download the Tivoli Policy Director Web Portal Manager, Version 3.7.1, from the following Web site:

http://www-internal.tivoli.com/secure/support/downloads/secureway/policy_dir/downloads.html

This appendix contains information about how to perform the following:

Logging into Tivoli Policy Director Web Portal Manager

Perform the following steps to log in to the Tivoli Policy Director Web Portal Manager:

1. From the Tivoli SecureWay Policy Director logon screen, which is shown in Figure 30 on page 82, enter your administrator user ID in the **User ID** field.

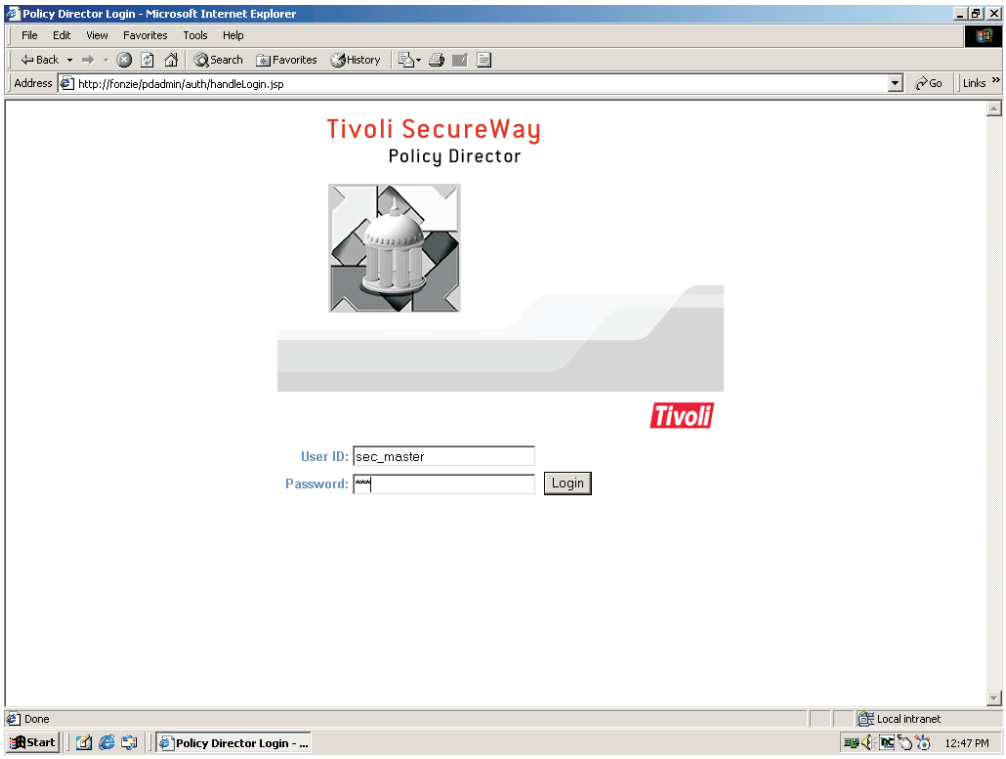

Figure 30. The Tivoli Policy Director logon screen

2. Enter your administrator password in the **Password** field.
3. Click **Login** to go to the Tivoli Policy Director Web Portal Manager screen. See Figure 31 on page 83.

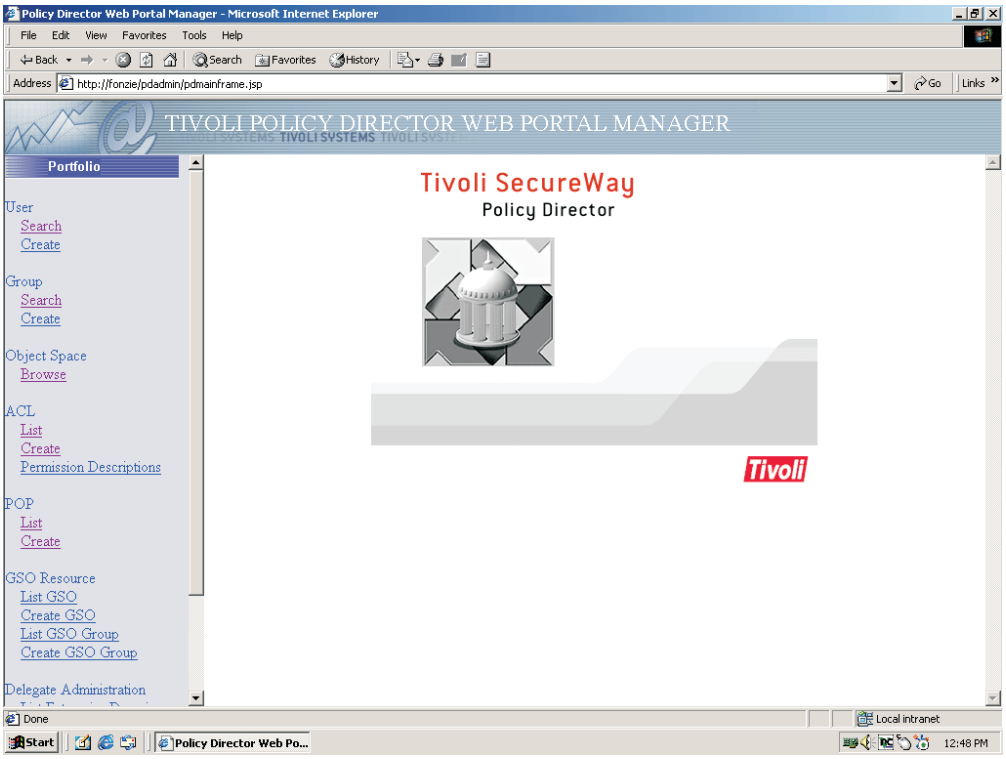

Figure 31. The Tivoli Policy Director Web Portal Manager screen

Browsing the Tivoli Policy Director Object Space

Perform the following steps to view the contents of the Tivoli Policy Director object space.

1. Under **Object Space** in the left-hand panel of the Tivoli Policy Director Web Portal Manager screen, click **Browse** to go to the Browse Object Space screen. See Figure 32 on page 84.

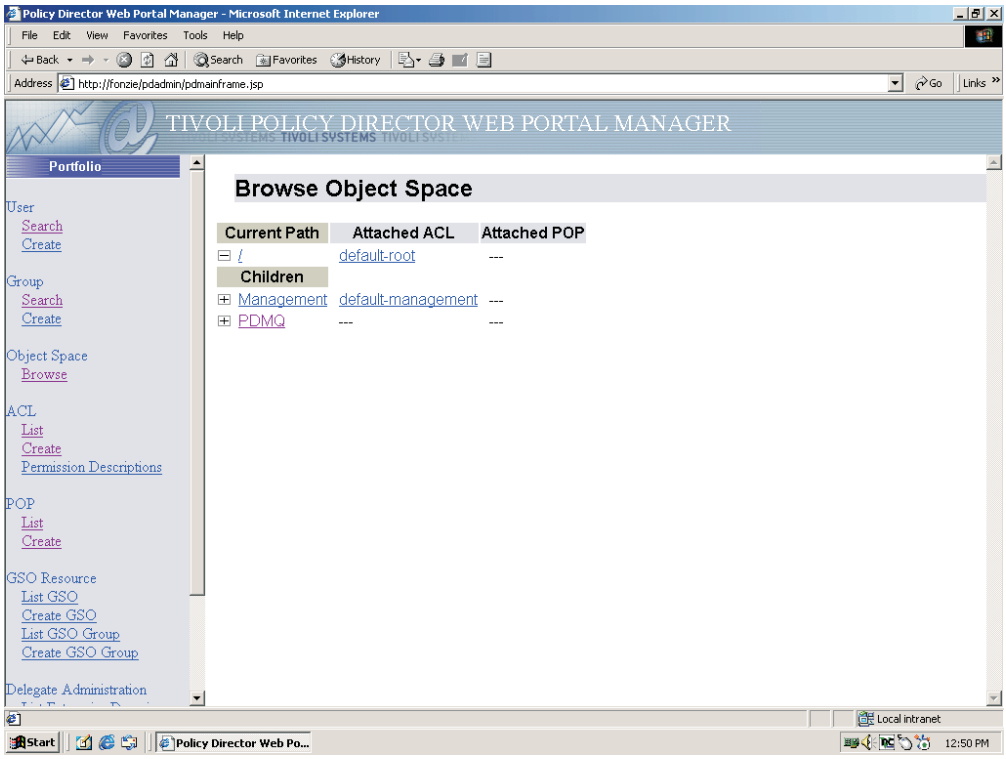

Figure 32. The Browse Object Space screen

2. In the main panel, click **+** to the left of **PDMQ** to expand it and display the existing Tivoli Policy Director for MQSeries queue managers.

Creating and Attaching an Access Control List for the Error Handling Queue

Perform the following steps to create an access control list (ACL) and attach it to the error handling queue:

1. Begin on the Tivoli Policy Director Web Portal Manager screen. See Figure 33 on page 85.

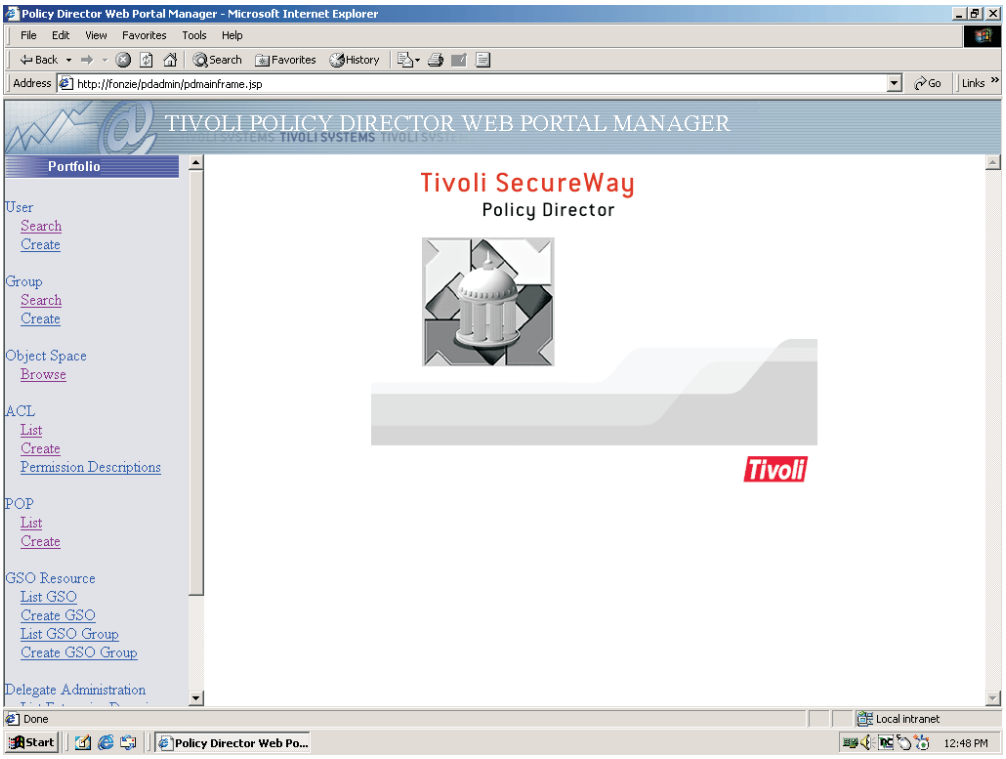

Figure 33. The Tivoli Policy Director Web Portal Manager screen

2. Under **ACL** in the left-hand panel of the screen, click **Create**. The Create ACL screen is displayed. See Figure 34.

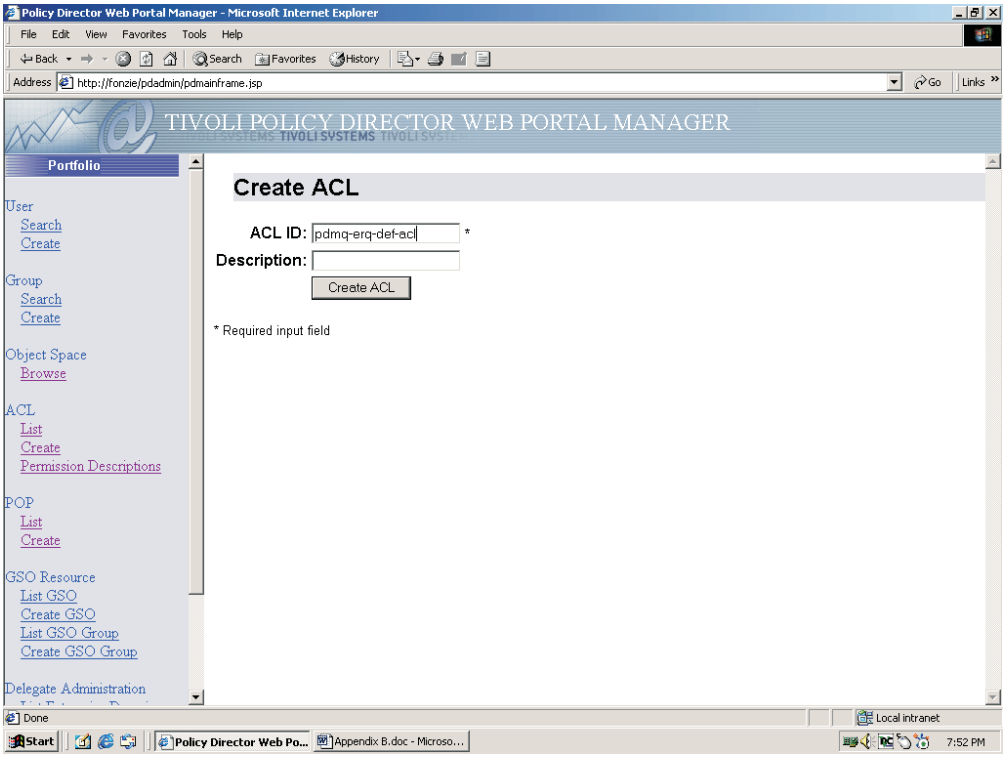

Figure 34. Create ACL screen

3. Enter the name of the ACL you need to attach in the **ACL ID** field.
4. Click **Create ACL**. The ACL Properties screen is displayed. See Figure 35.

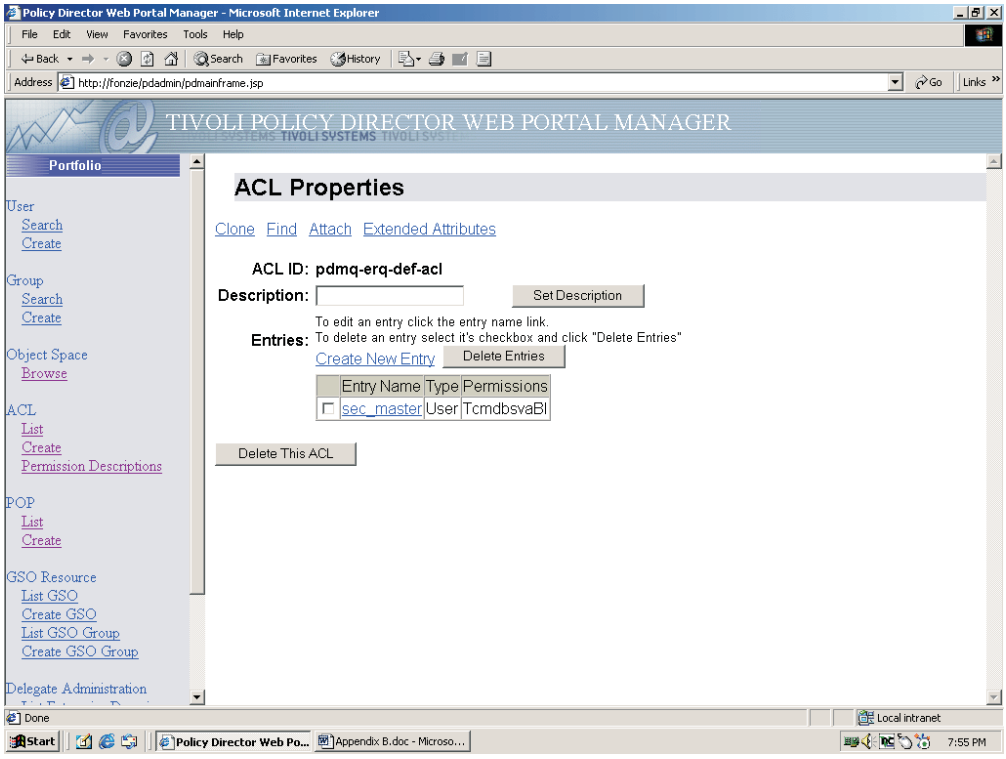

Figure 35. ACL Properties screen

5. Click **Create New Entry**. The Create ACL Entry screen is displayed. See Figure 36 on page 87.

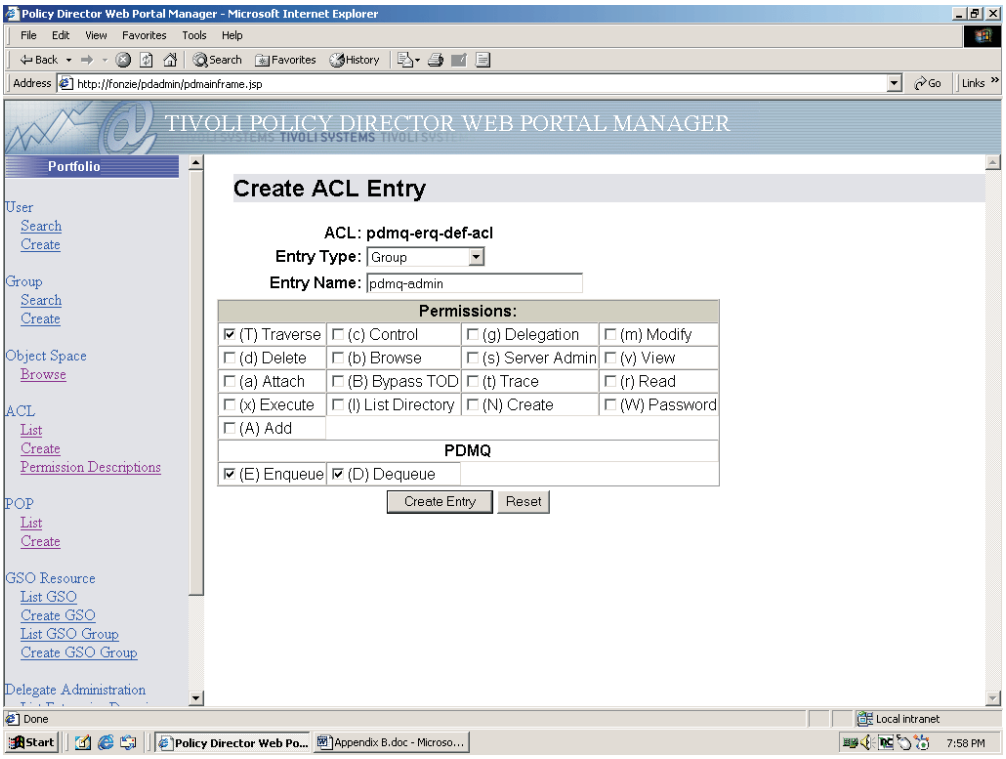

Figure 36. Create ACL Entry screen

6. Verify that **Group** is selected in the **Entry Type** field.
7. Specify `pdmq-admin` in the **Entry Name** field.
8. Select permissions **(T) Traverse**, **(E) Enqueue**, and **(D) Dequeue**.
9. Click **Create Entry** to return to the ACL Properties screen.
10. Review the information on the ACL Properties screen. See Figure 37 on page 88.

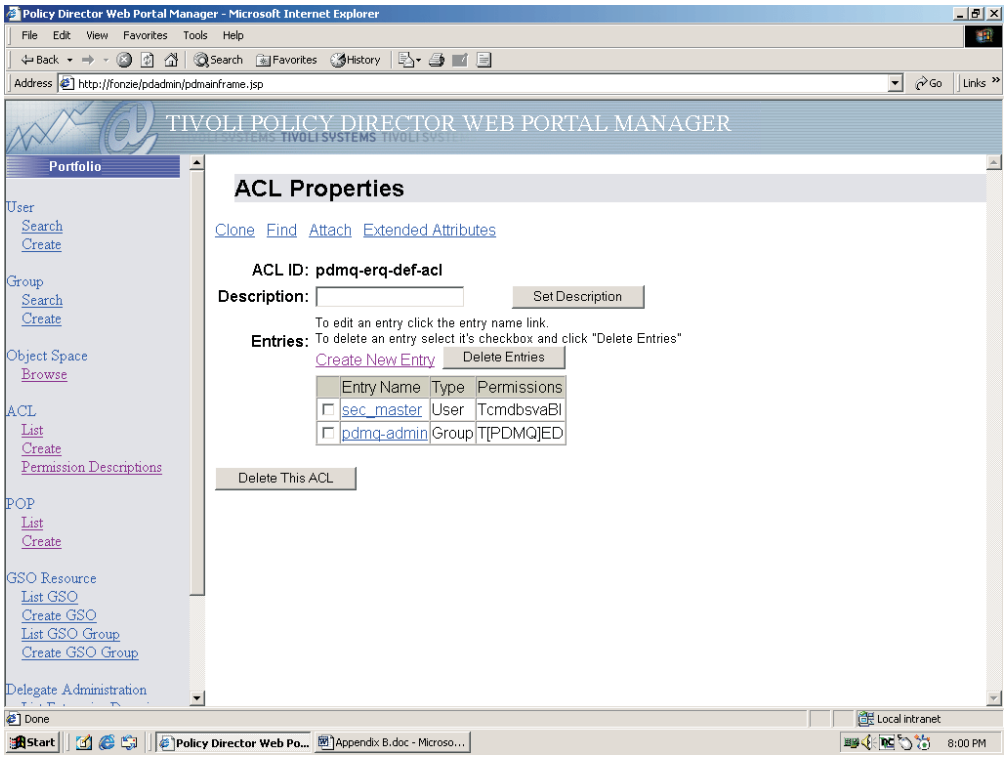

Figure 37. ACL Properties screen

11. Click **Create New Entry** to return to the Create ACL Entry screen. See Figure 38.

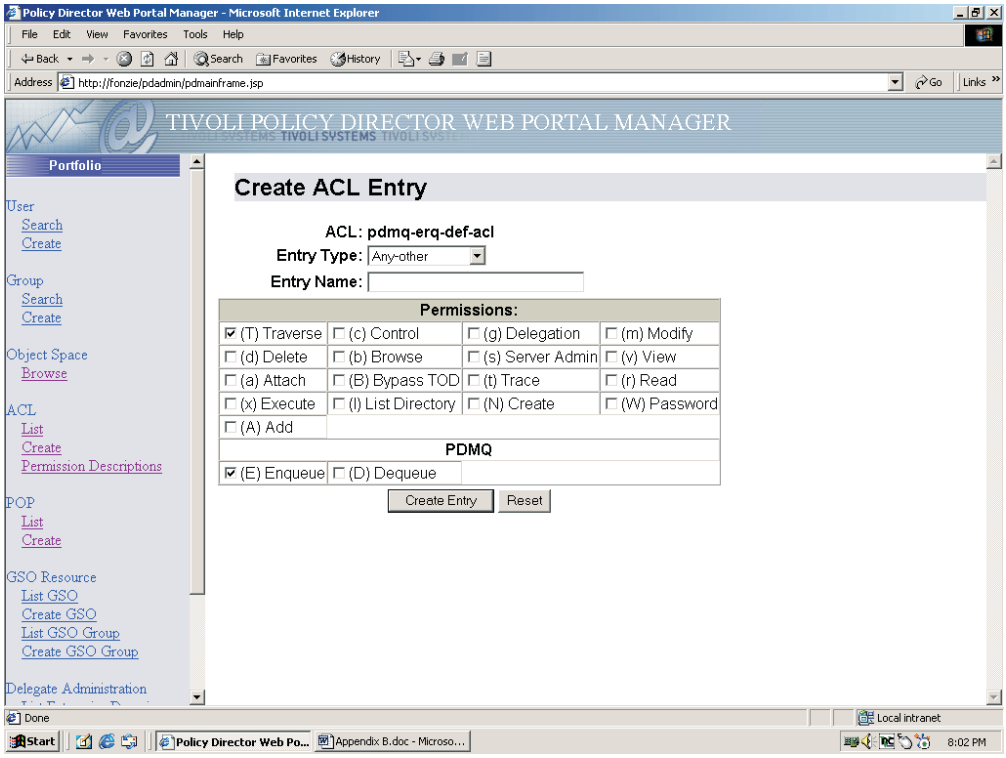

Figure 38. Create ACL Entry screen

12. Select **Any-other** from the drop-down list in the **Entry Type** field.
13. Select permissions **(T) Traverse** and **(E) Enqueue**.
14. Click **Create Entry** to return to the ACL Properties screen.
15. Review the information on the screen. See Figure 39.

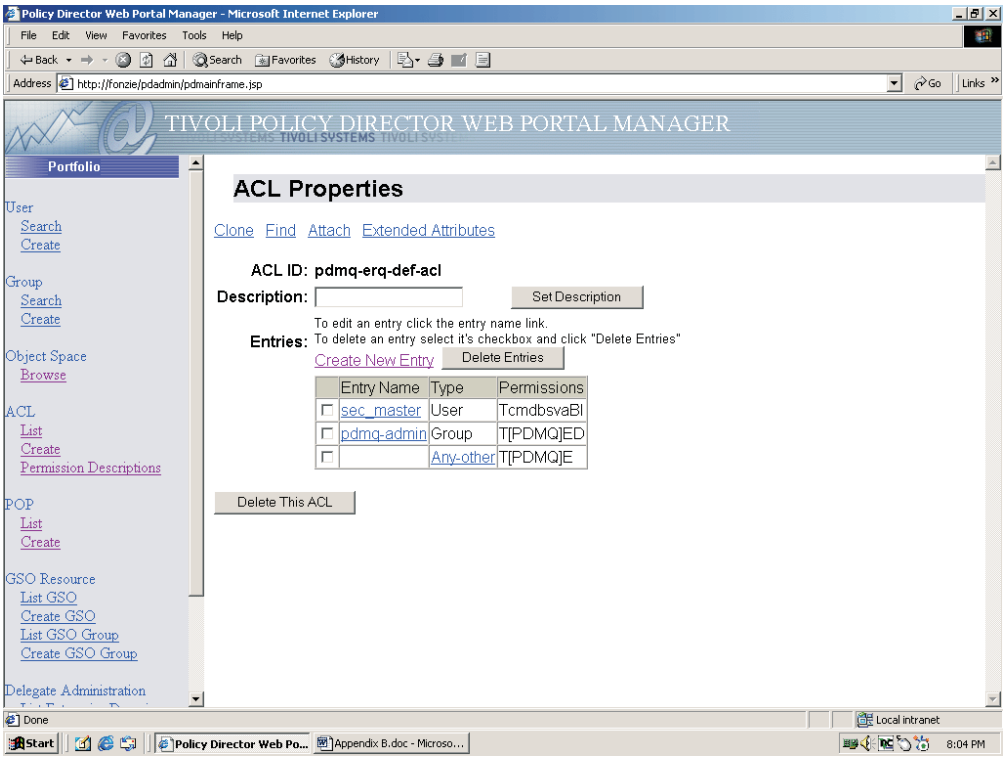

Figure 39. ACL Properties screen

16. At the top of the screen, click **Attach** to go to the Attach ACL screen. See Figure 40 on page 90.

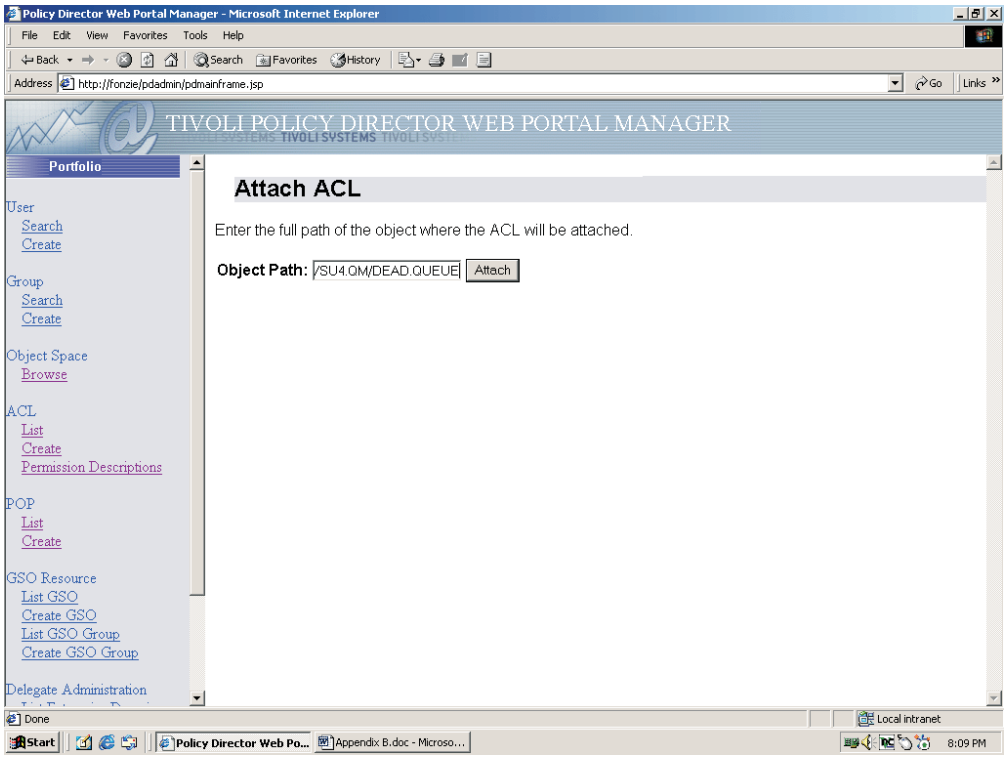

Figure 40. Attach ACL screen

17. Enter the path of the error handling queue in the **Object Path** field.
18. Click **Attach** to attach this ACL to the object space.
19. Browse the object space to verify that the ACL is attached. (See "Browsing the Tivoli Policy Director Object Space" on page 83 for instructions on how to browse the object space.)

    Note: In the example shown in Figure 41 on page 91, the ACL `pdmq-erq-def-acl` is created and attached to the error handling queue DEAD.QUEUE in the queue manager SSFVSU4.QM.

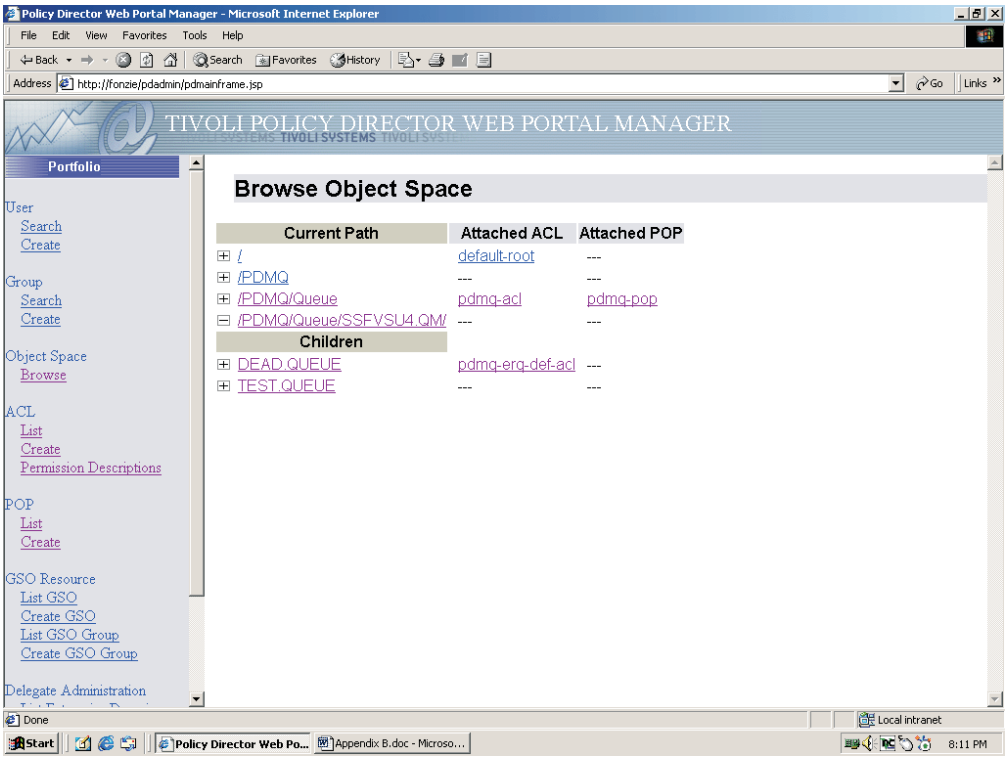

Figure 41. Verify that the ACL Is attached

Creating and Attaching the Protected Object Policy for the Error Handling Queue

Perform the following steps to create a protected object policy (POP) and attach it to the error handling queue:

1. Begin on the Tivoli Policy Director Web Portal Manager screen. See Figure 42 on page 92.

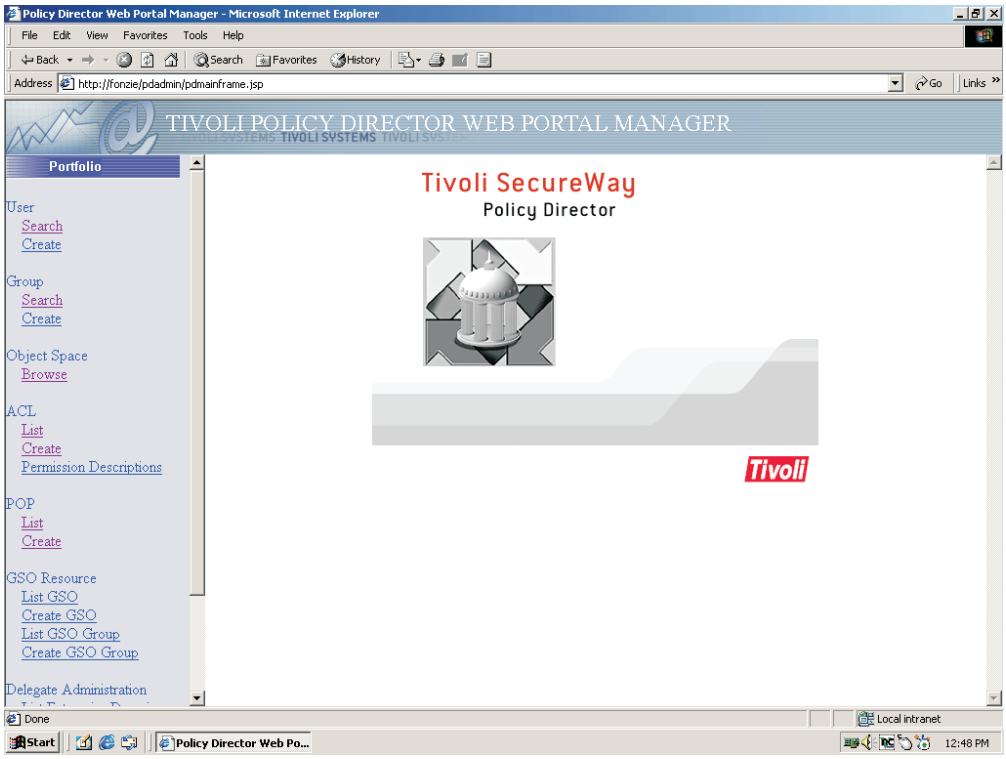

Figure 42. Tivoli Policy Director Web Portal Manager screen

2. Under **POP** in the left-hand panel of the screen, click **Create**. The Create Protected Object Policy screen is displayed. See Figure 43 on page 93.

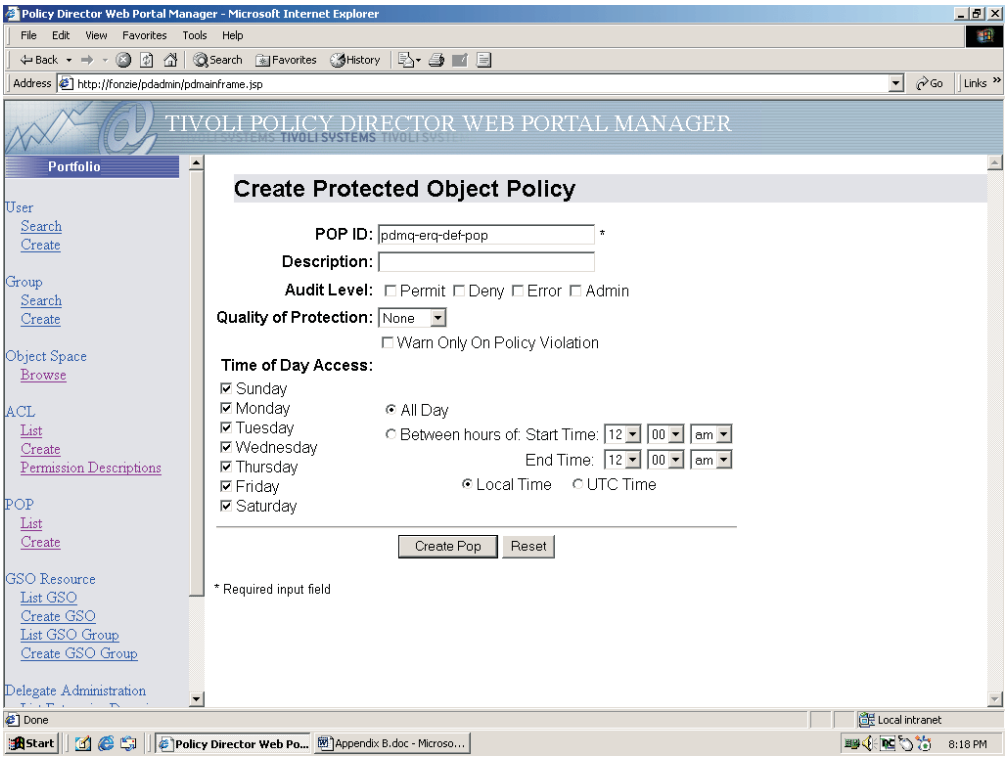

Figure 43. Create Protected Object Policy screen

3. Enter the name of the protected object policy that you need to create in the **POP ID** field.
4. Ensure that the **Audit Level** is set to **None** (no **Audit Level** boxes are selected).
5. Ensure that the **Quality of Protection** is set to **None**.
6. Click **Create Pop**. The Create Protected Object Policy screen is displayed. See Figure 44 on page 94.

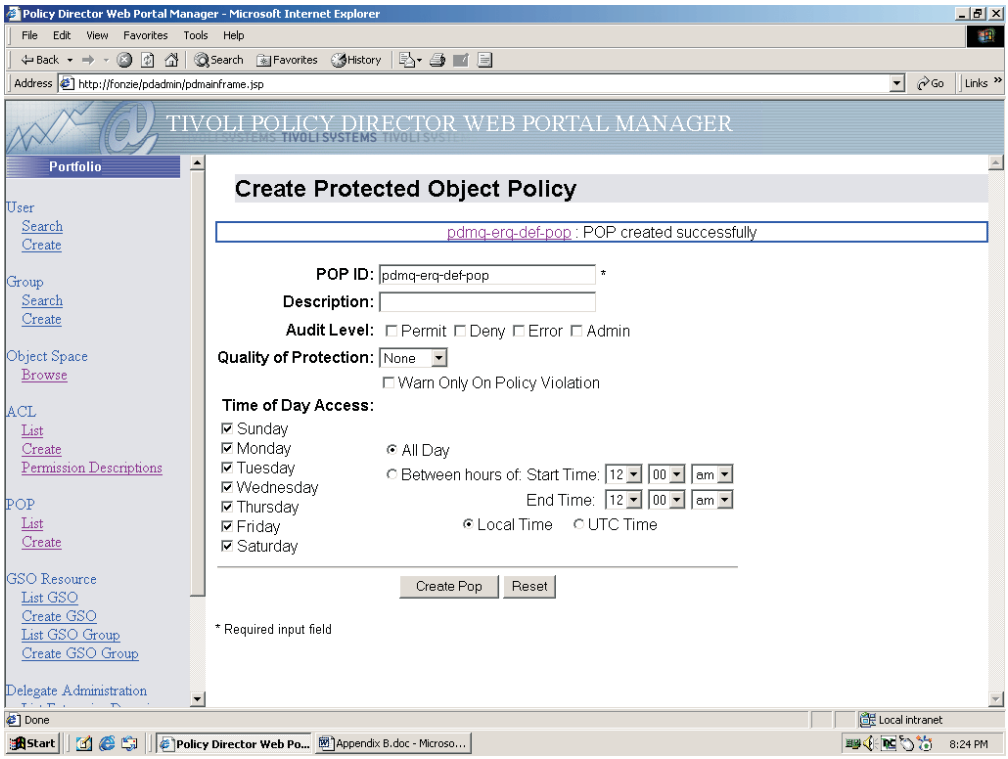

Figure 44. Create Protected Object Policy screen

7. Review the information message at the top of the screen, *pop_ID* `: POP created successfully.`

   Note: In the example shown, you create the POP called `pdmq-erq-def-pop`.
8. Under **POP** in the left–hand panel of the screen, click **List**. The POP List screen is displayed. See Figure 45 on page 95.

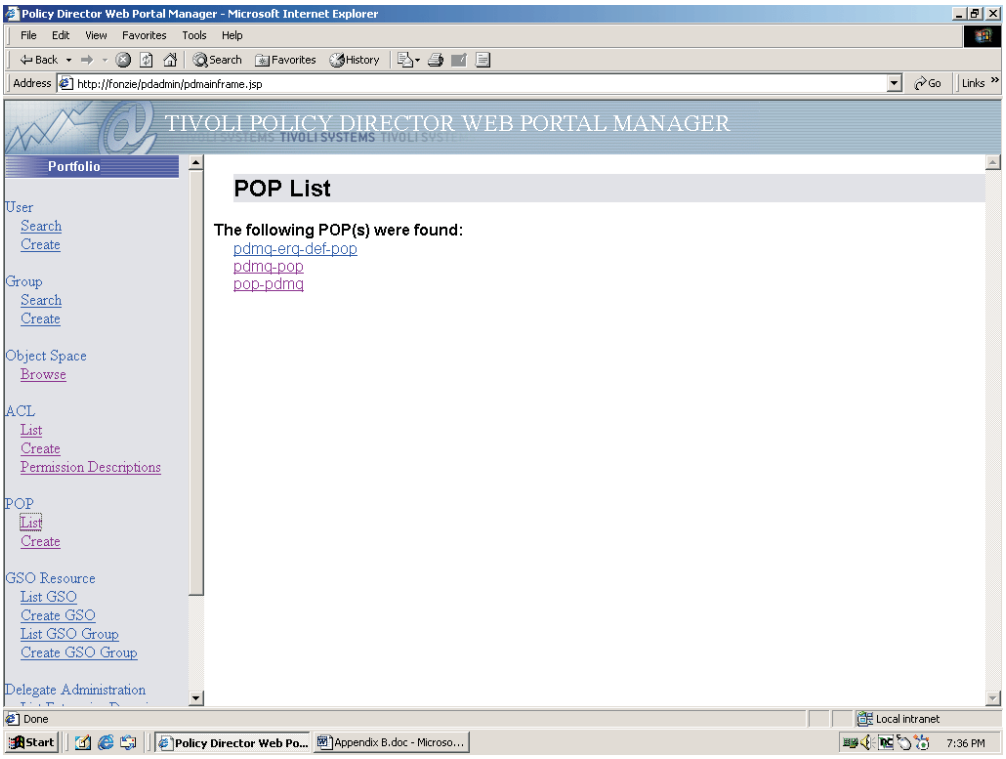

Figure 45. POP List screen

9. Click the name of the POP that you just created (`pdmq-erq-def-pop` in the example). The POP Properties screen for the selected POP is displayed. See Figure 46 on page 96.

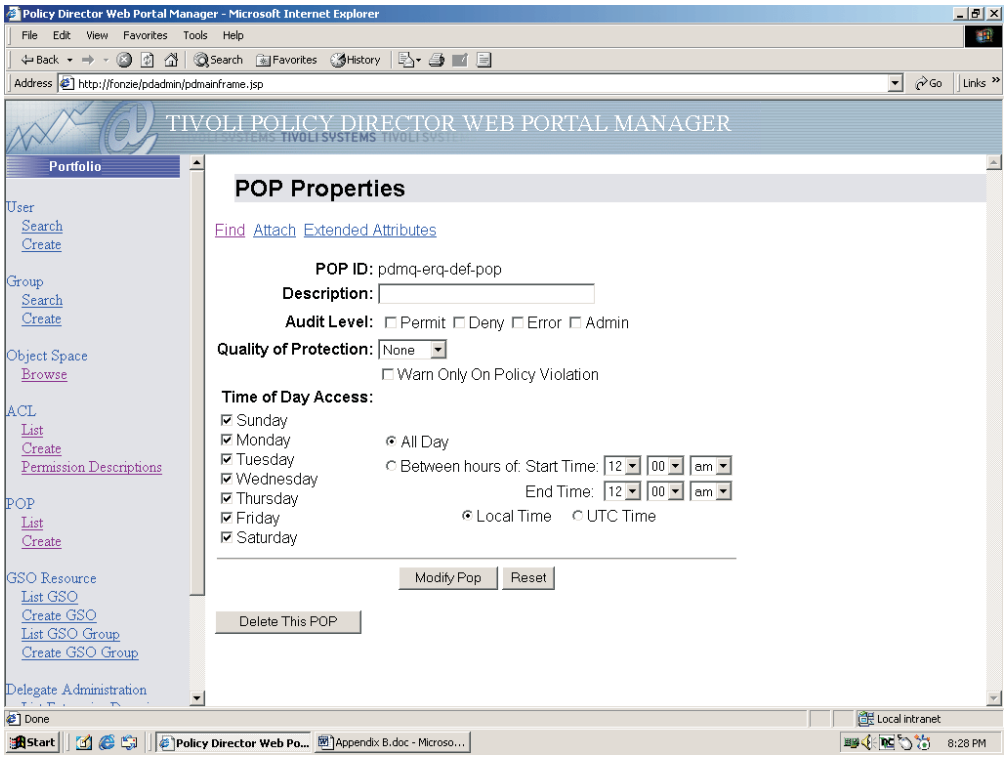

Figure 46. POP Properties screen

10. At the top of the screen, click **Attach** to attach this POP to the object space. The Attach POP screen displays. See Figure 47.

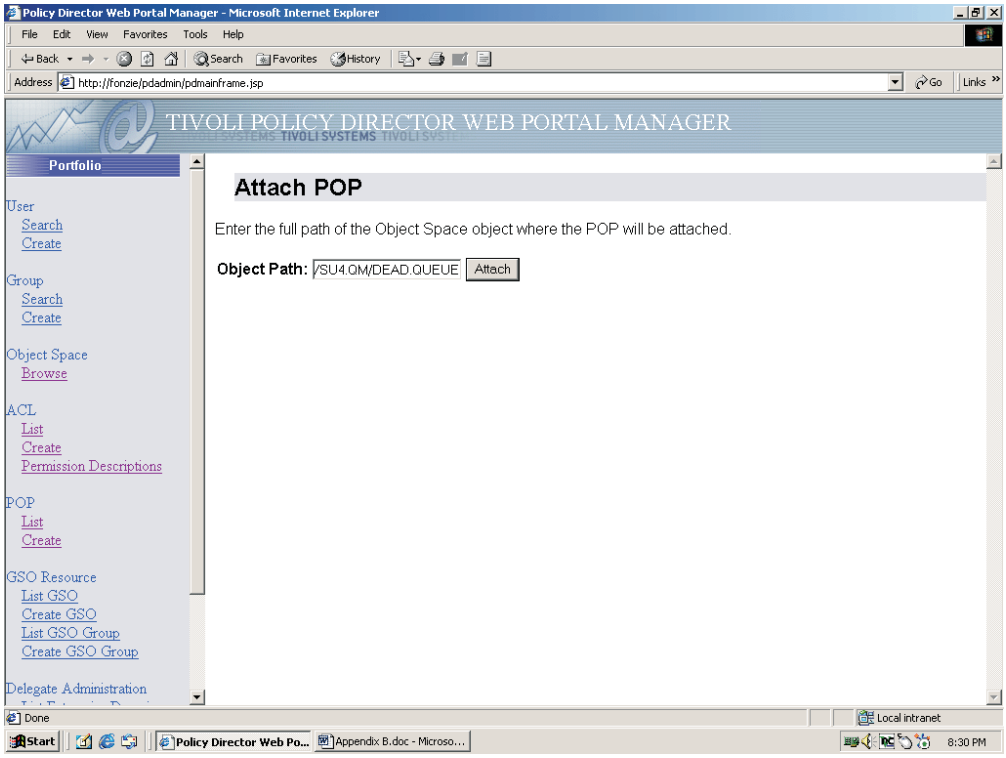

Figure 47. Attach POP screen

11. Enter the path of the error handling queue in the **Object Path** field.
12. Click **Attach**. The POP is attached to the specified object.
13. Browse the object space to verify that the POP is attached to the object. See Figure 48.

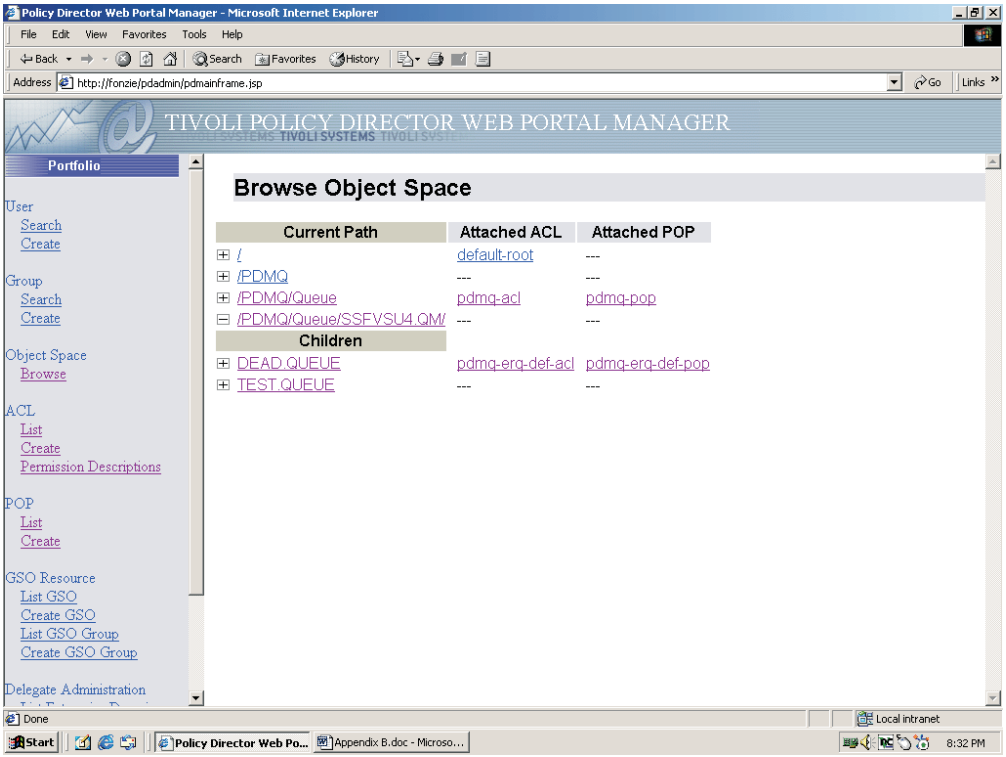

Figure 48. Verify that POP is attached

Note: In the example shown, the POP `pdmq-erq-def-pop` is attached to the error handling queue DEAD.QUEUE in the queue manager SSFVSU4.QM.

Configuring /PDMQ/Queue/*queue_manager*

Perform the following steps to create new attributes for `/PDMQ/Queue/queue_manager`:

Note: In the example shown *queue_manager* is SSFVSU4.QM, and the queue `DEAD.QUEUE` is set up to be the error handling queue for this queue manager.

1. Start from the Browse Object Space screen, as shown in Figure 49 on page 98. (See "Browsing the Tivoli Policy Director Object Space" on page 83 for instructions on displaying this screen.)

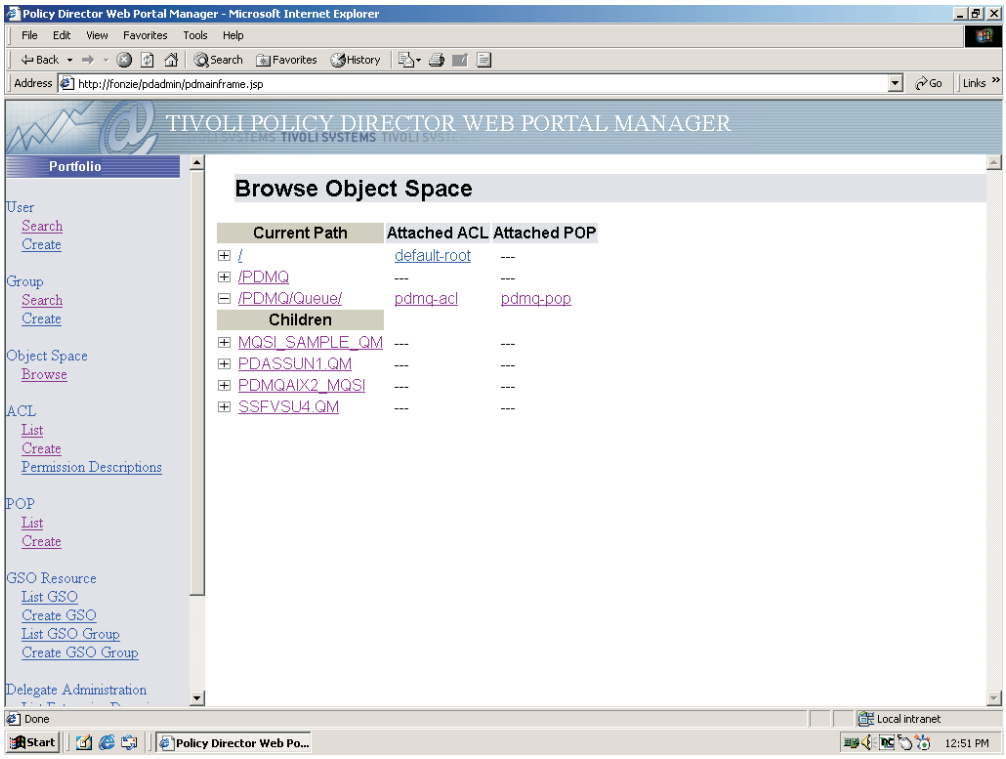

Figure 49. Browse Object Space screen

2. Click the name of the queue manager that you need to configure. The Protected Object Properties screen is displayed, and the indicated **Protected Object ID** is `/PDMQ/Queue/`*`queue_manager`*. See Figure 50 on page 99 and Figure 51 on page 99.

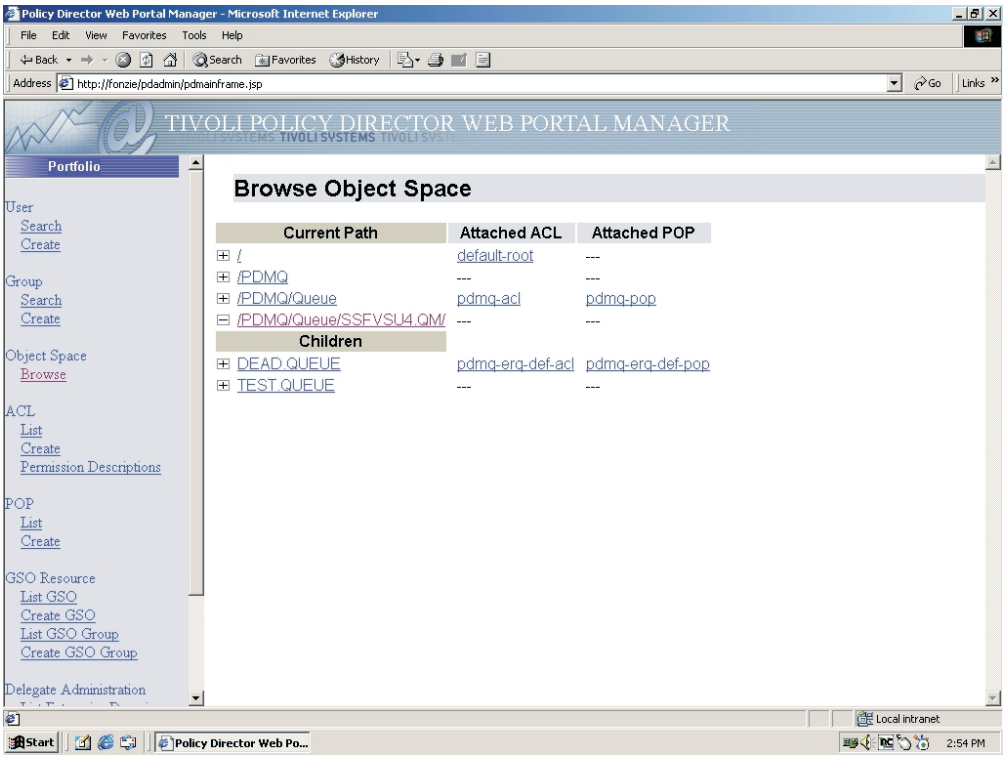

Figure 50. Expand queue manager SSFVSU4.QM

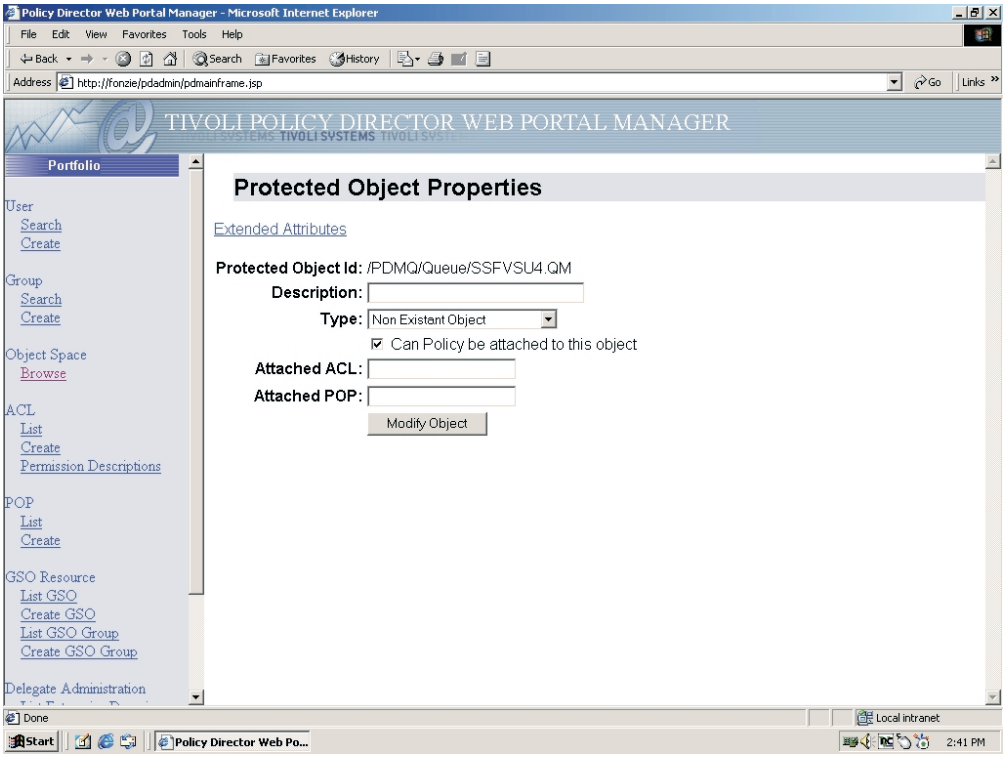

Figure 51. Protected object properties for specified queue manager (SSFVSU4.QM)

3. In the main panel of the screen, click **Extended Attributes**. The Extended Attributes screen for the specified queue manager is displayed. See Figure 52.

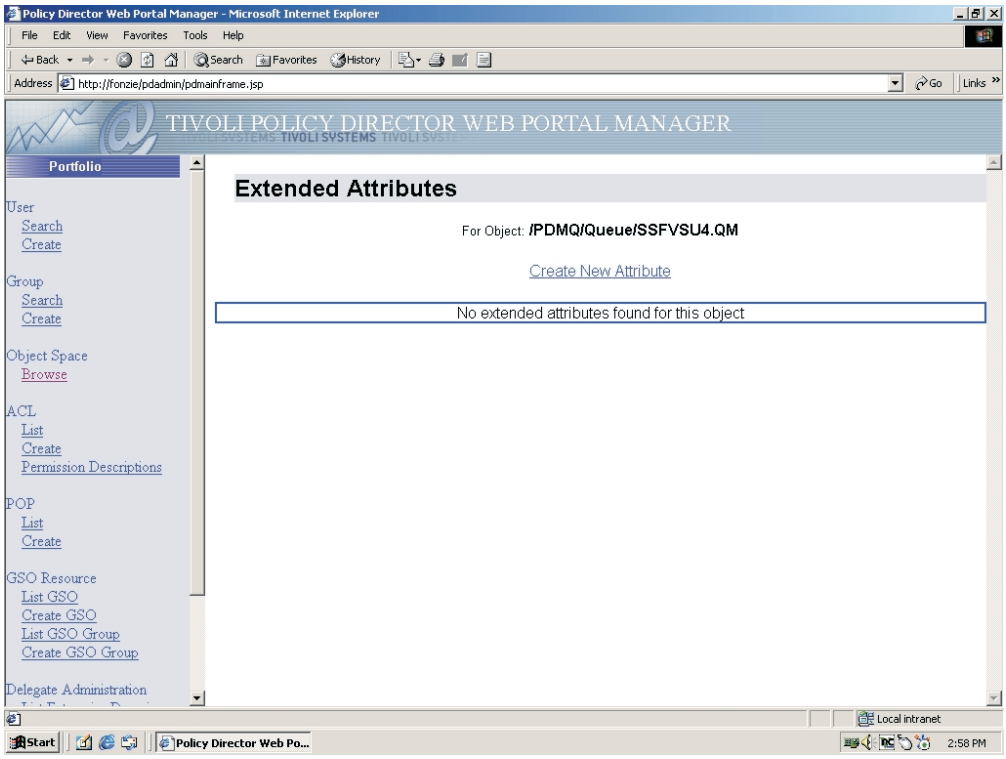

Figure 52. Extended attributes for specified queue manager (SSFVSU4.QM)

4. In the main panel of the screen, click **Create New Attribute**. The Create Extended Attribute screen is displayed. See Figure 53 on page 101.

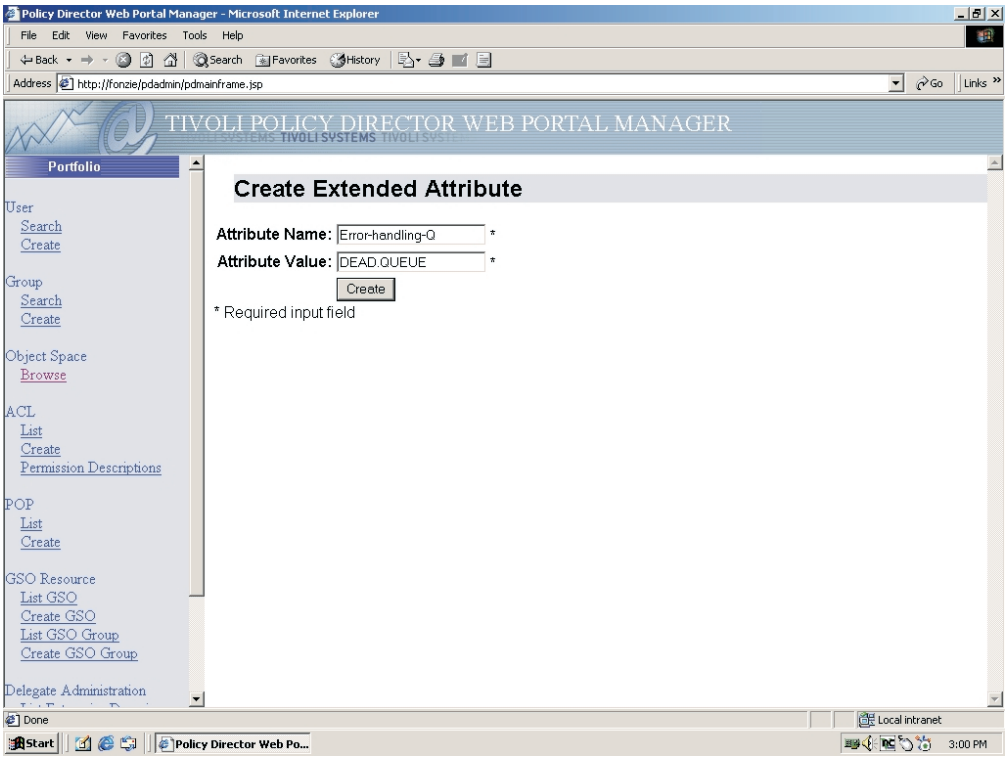

Figure 53. Create extended attribute for specified queue manager (SSFVSU4.QM)

5. Enter the name of the attribute in the **Attribute Name** field. In the example shown, the **Attribute Name** is `Error-handling-Q`.
6. Enter the name of the queue in the **Attribute Value** field. In the example shown, the **Attribute Value** is `DEAD.QUEUE`.
7. Click **Create** to create the attribute and return to the Extended Attributes screen.
8. Review the information on the screen under **Current Attributes** that describes the change you have made. See the example in Figure 54 on page 102.

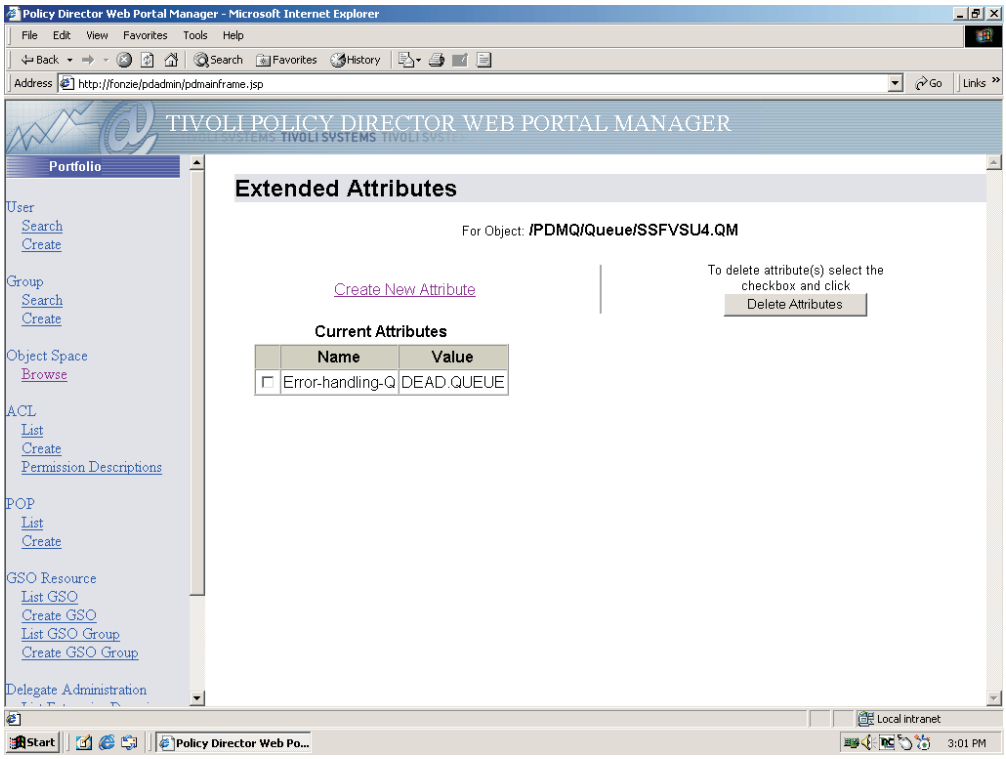

Figure 54. Error handling queue assigned for specified queue manager (SSFVSU4.QM)

Configuring /PDMQ/Queue/*queue_manager*/*queue*

Perform the following steps to configure a queue in the object space:

1. Browse the object space (see "Browsing the Tivoli Policy Director Object Space" on page 83 for instructions on how to display this screen). See Figure 55 on page 103.

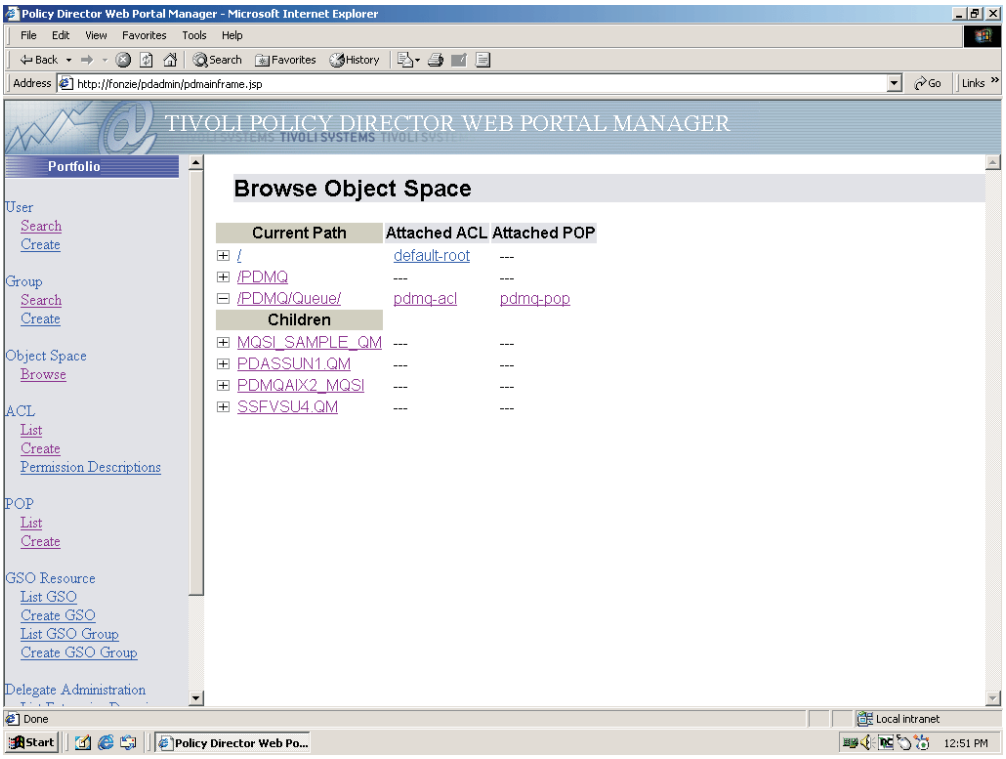

Figure 55. Browse Object Space screen

2. On the Browse Object Space screen, expand the queue manager that contains the queue that you need to configure. The contents of the specified queue manager are displayed. See Figure 56 on page 104.

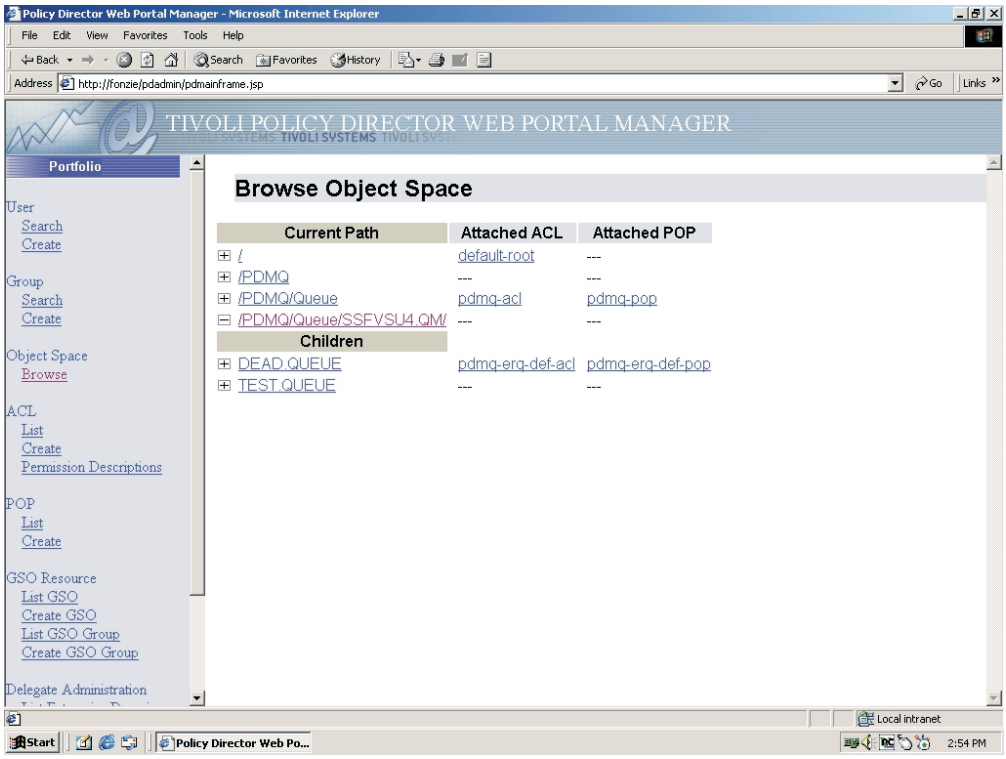

Figure 56. Browse Object Space screen for specified queue manager (SSFVSU4.QM)

3. Click the name of the queue that you need to configure. The Protected Object Properties screen is displayed. See Figure 57.

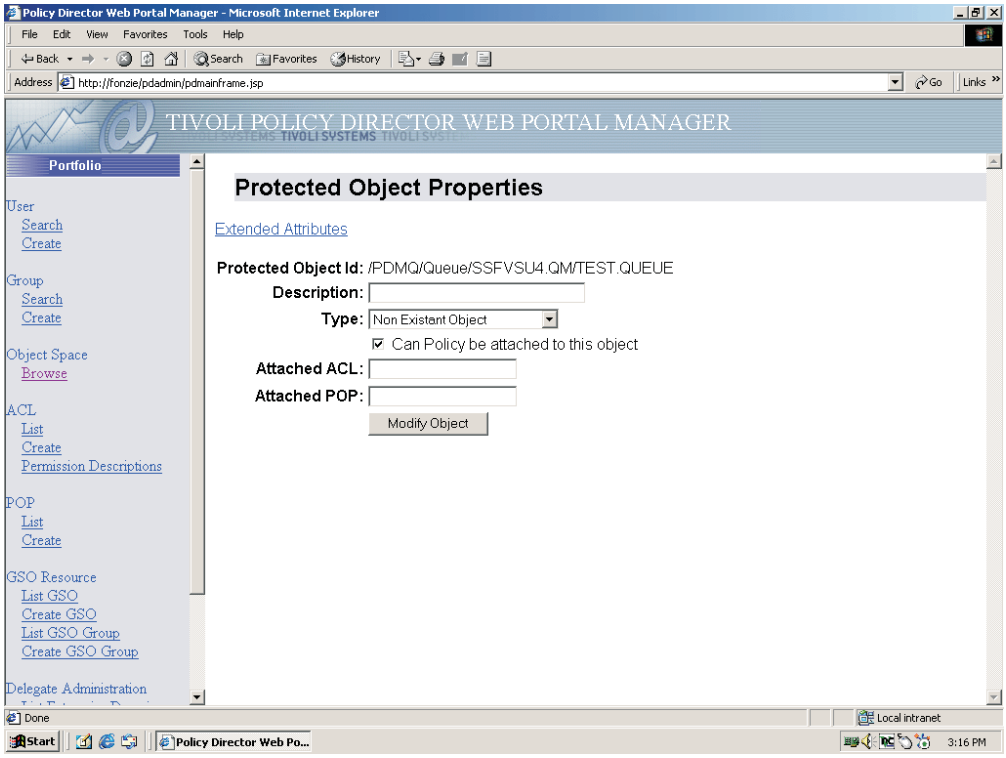

Figure 57. Protected object properties for specified queue (TEST.QUEUE)

4. Click **Extended Attributes**. The Extended Attributes screen is displayed. See Figure 58.

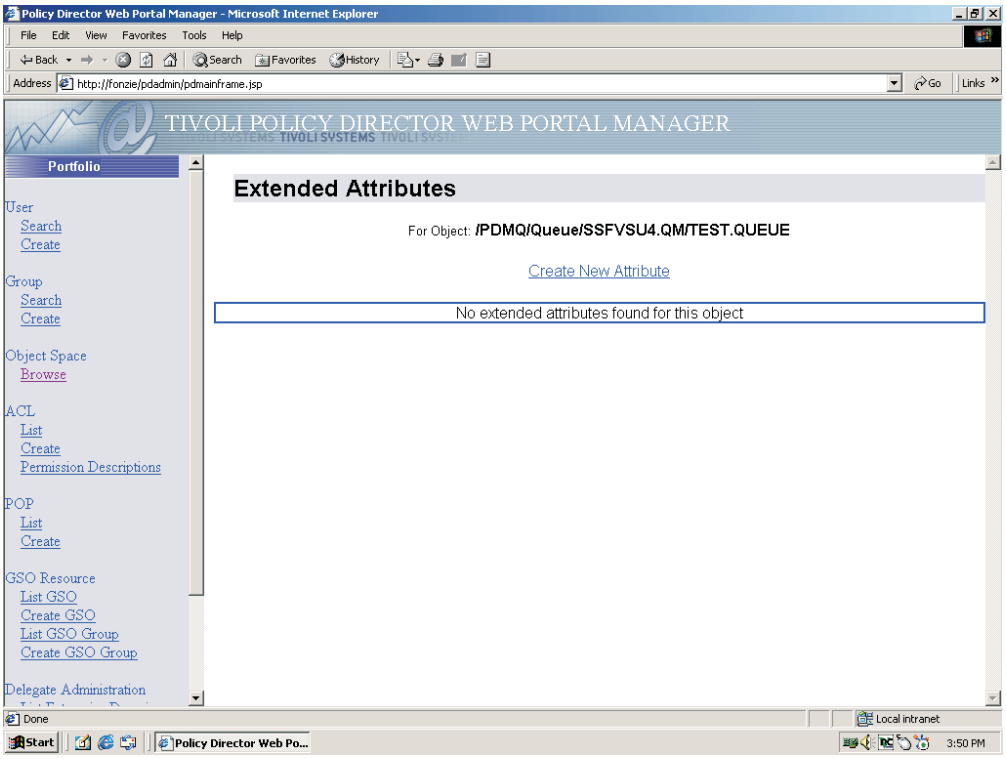

Figure 58. Extended attributes for specified queue (TEST.QUEUE)

5. Review the information on the screen.
6. Click **Create New Attribute**. The Create Extended Attributes screen is displayed. See Figure 59 on page 106.

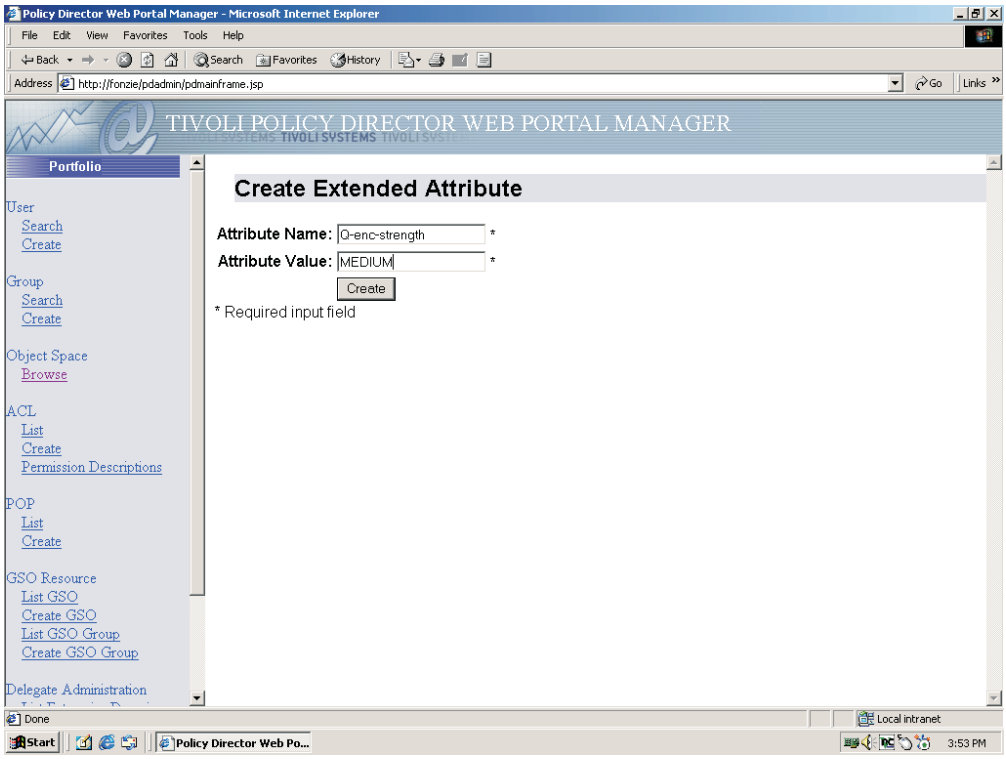

Figure 59. Create Extended Attributes screen for selected queue (TEST.QUEUE)

7. Enter the name of the attribute that you want to assign to the queue in the **Attribute Name** field.
8. Click **Create** to return to the Extended Attributes screen. See Figure 60 on page 107.

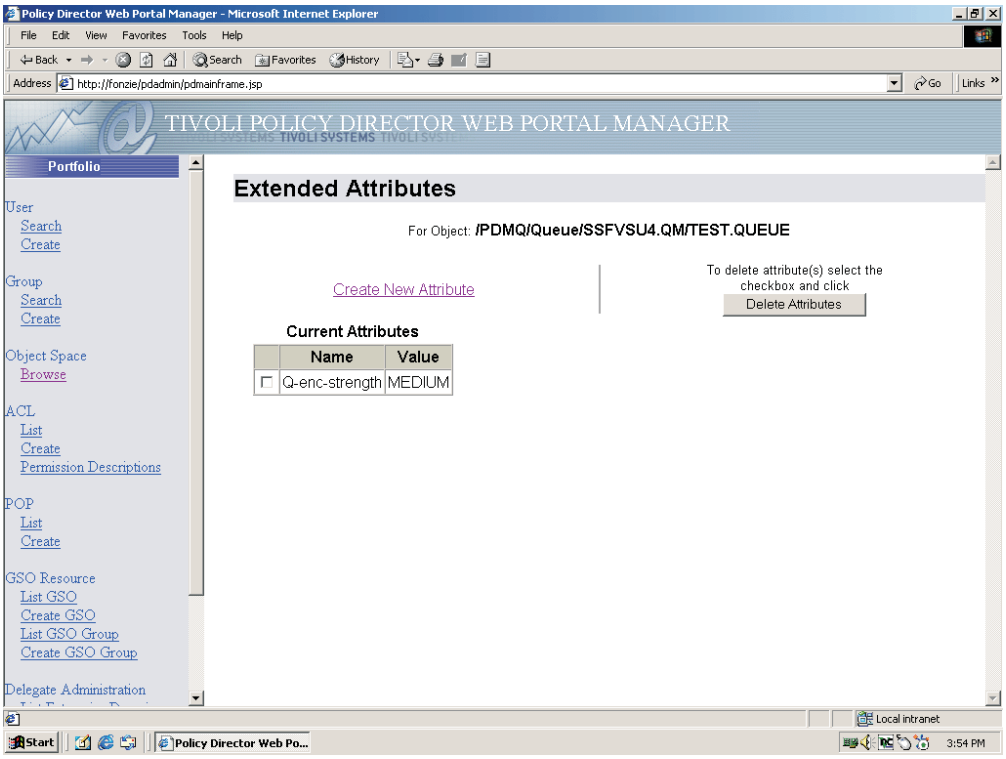

Figure 60. Extended Attributes screen

9. Review the information on the screen.
10. If you need to create another extended attribute, repeat Steps 6–10. When you have created all the necessary attributes, proceed to the next section.

    Note: In the example shown in Figure 61 on page 108, the following attributes and values are created for the queue TEST.QUEUE in the queue manager SSFVSU4.QM:

    ```
    Q-enc-strength   MEDIUM
    Q-sig-algorithm   MD5
    Q-recipients   CN-tivolipki pdmq cert;O=Tivoli;C=US
    ```

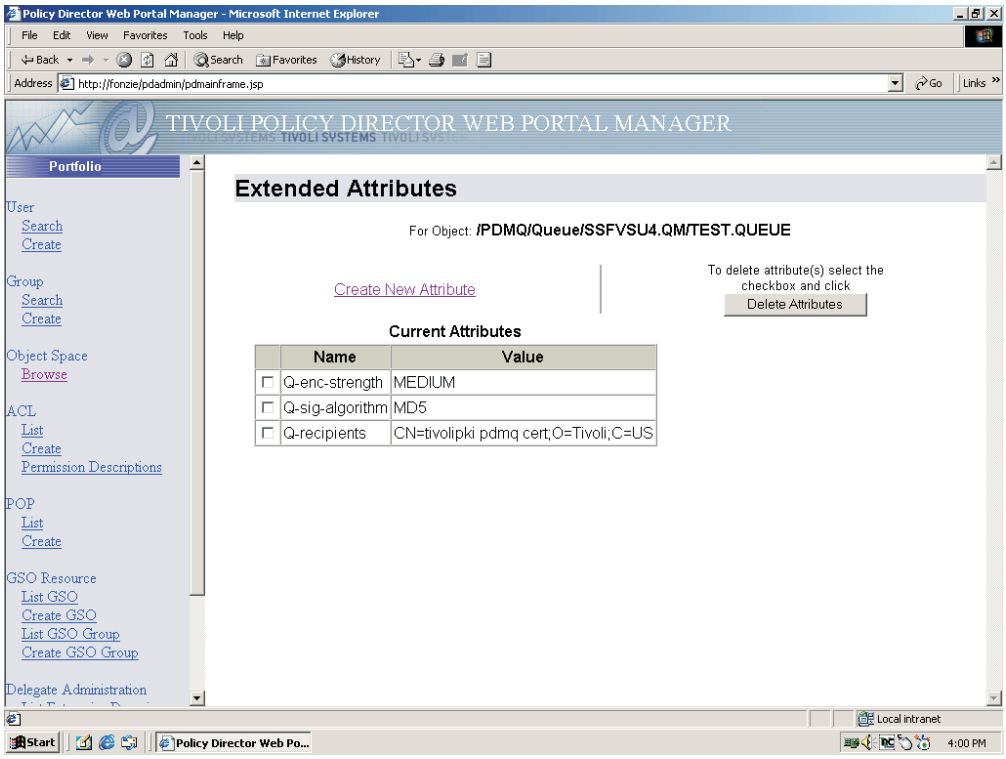

Figure 61. Extended attributes created for specified queue (TEST.QUEUE)

Specifying Authorization for Tivoli Policy Director for MQSeries Operations

Perform the following steps to grant authorization for a particular user to perform Tivoli Policy Director for MQSeries operations:

1. Begin with the Tivoli Policy Director Web Portal Manager screen. See Figure 62 on page 109.

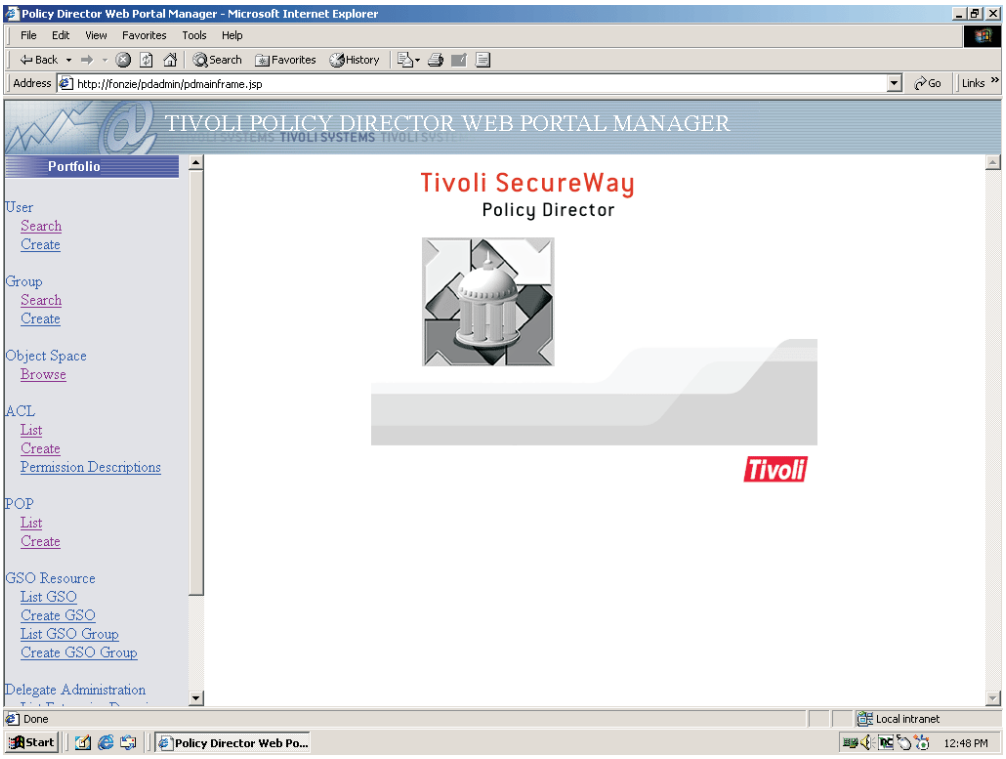

Figure 62. Tivoli Policy Director Web Portal Manager screen

2. Under **ACL** in the left-hand panel of the screen, click **Create**. The Create ACL screen displayed. See Figure 63.

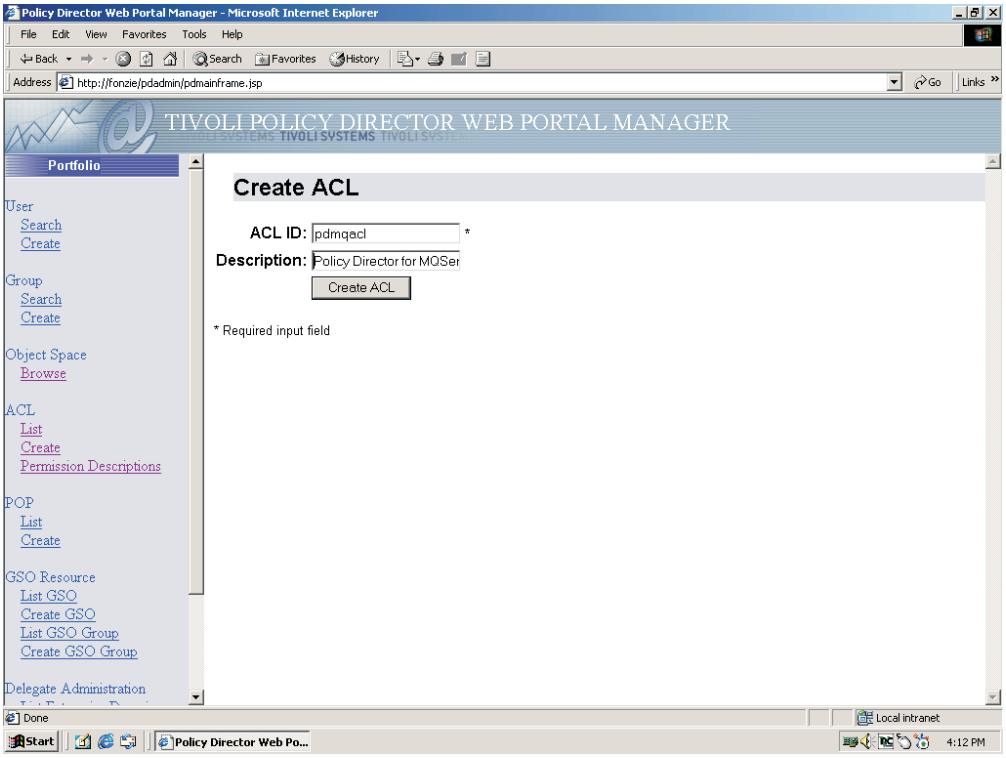

Figure 63. Create ACL screen

3. In the ACL ID field, specify the name of the ACL for which you need to authorize a user.
4. Click **Create ACL**. The ACL Properties screen is displayed. See Figure 64.

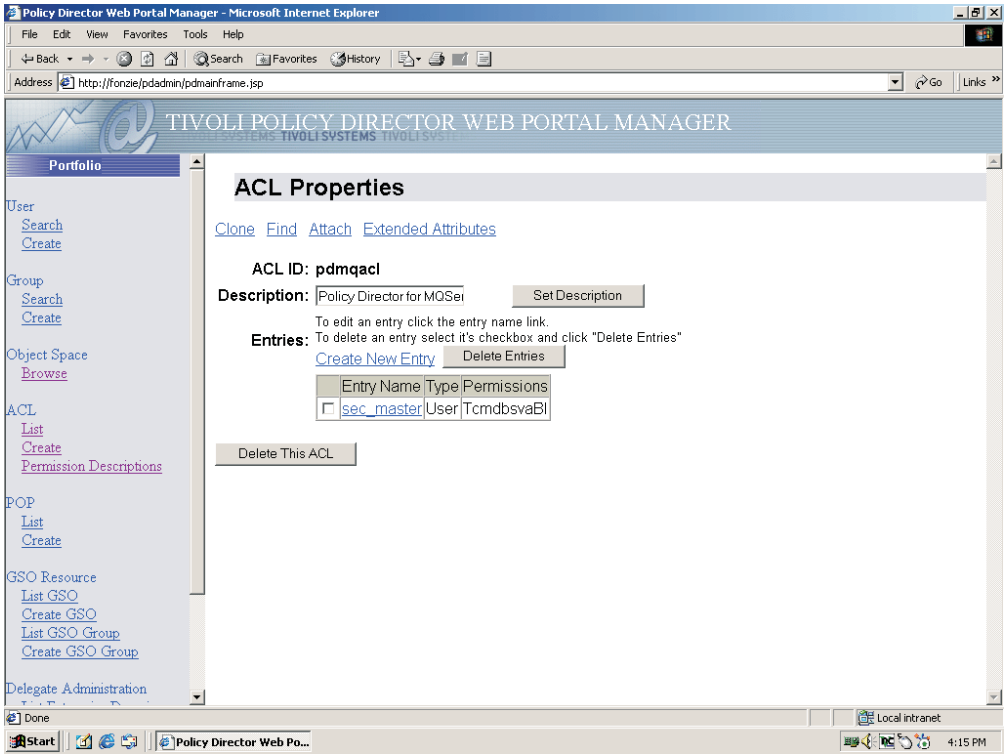

Figure 64. ACL Properties

5. Click **Create New Entry**. The Create ACL Entry screen is displayed. See Figure 65 on page 111.

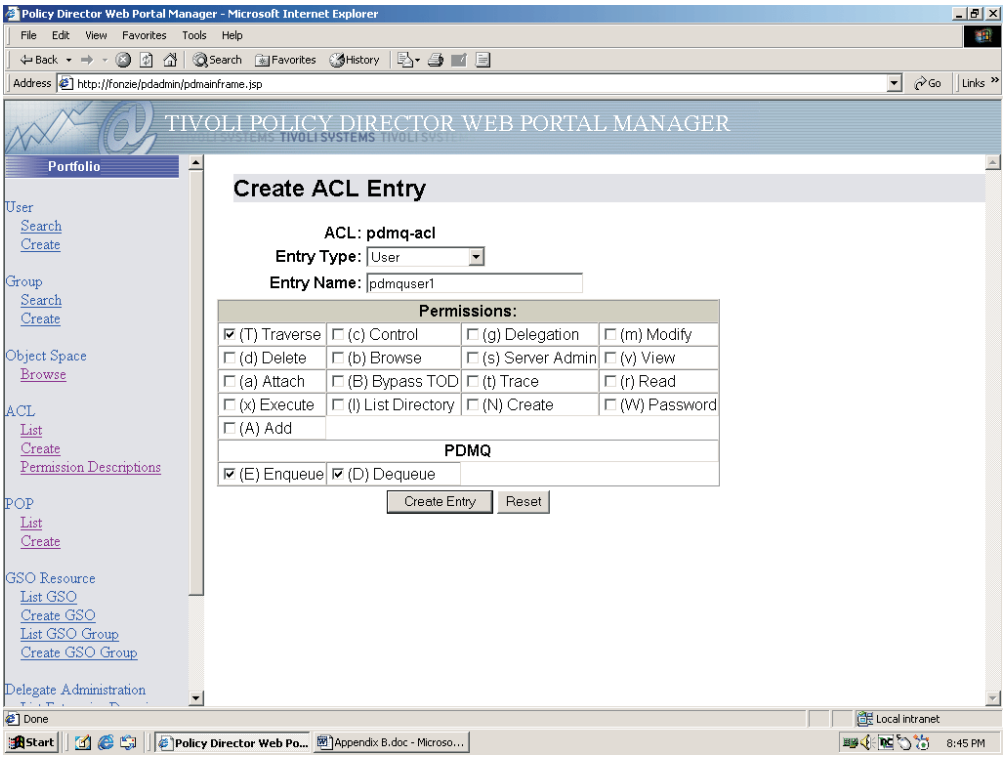

Figure 65. Create ACL entry screen

6. Select **User** from the drop-down list in the **Entry Type** field.
7. Enter the user ID of the individual that needs authorization in the **Entry Name** field.
8. Check the boxes next to the permissions and operations that the specified user is to be authorized to perform.
9. Click **Create Entry**. The ACL Properties screen is displayed. See Figure 66 on page 112.

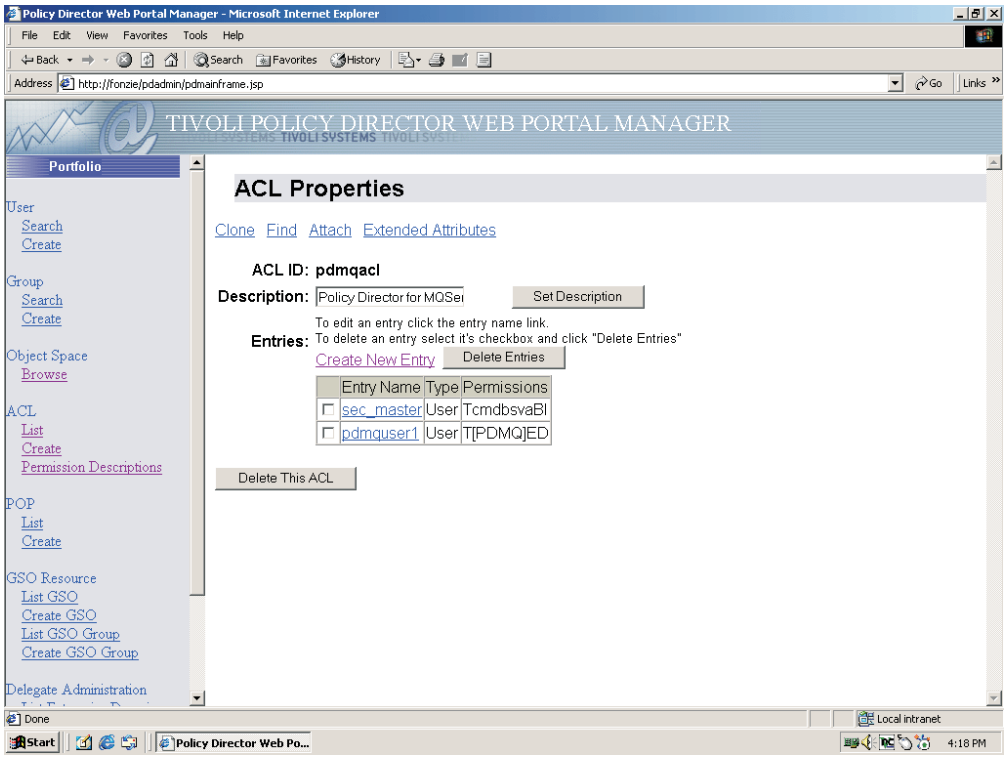

Figure 66. ACL Properties screen

10. Click **Attach** to attach this ACL to the Tivoli Policy Director object space. The Attach ACL screen is displayed. See Figure 67 on page 113.

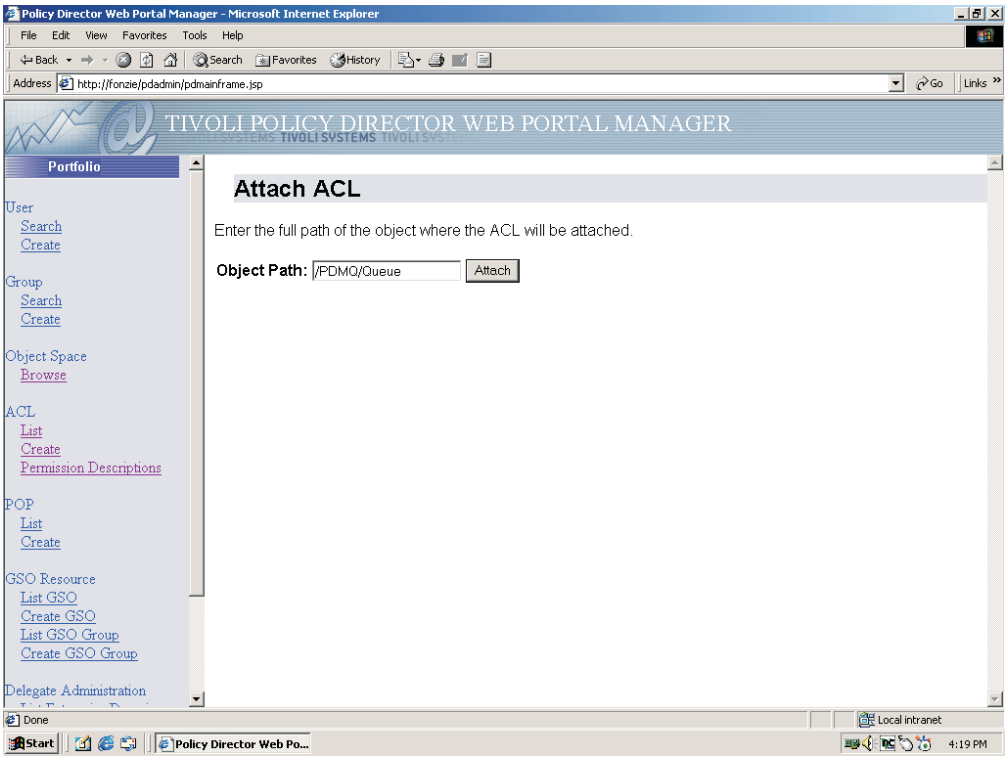

Figure 67. Attach ACL screen

11. Enter the path of the object to which you need to attach the ACL in the **Object Path** field.
12. Click **Attach**. The Browse Object Space screen is displayed. See Figure 68 on page 114.

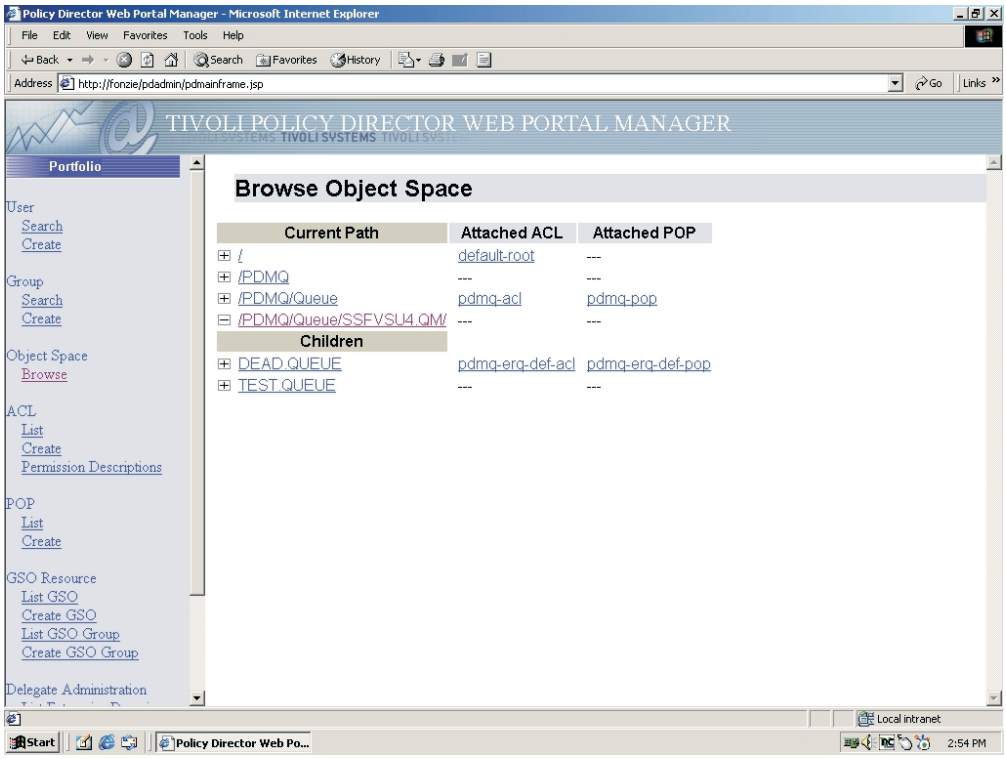

Figure 68. Verify that the ACL Is attached

13. Review the information on the screen to verify that the ACL is attached.

    Note: In the example shown, the user `pdmquser1` is added to the ACL with the permission **(T) Traverse** and authorization to enqueue and dequeue messages. This ACL applies to all queues under the `/PDMQ/Queue` level of the object space.

Configuring the Protected Object Policy

Perform the following steps to configure the protected object policy:

1. Begin with the Tivoli Policy Director Web Portal Manager screen. See Figure 69 on page 115.

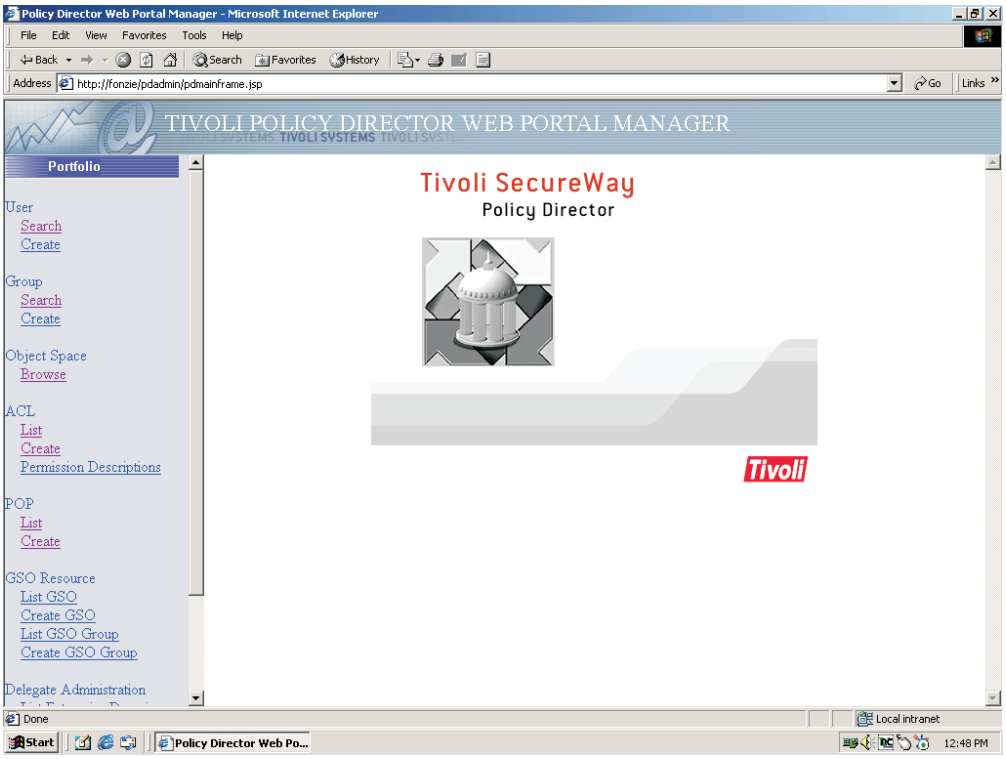

Figure 69. Tivoli Policy Director Web Portal Manager screen

2. Under **POP** in the left-hand panel of the screen, click **Create**. The Create Protected Object Policy screen is displayed. See Figure 70.

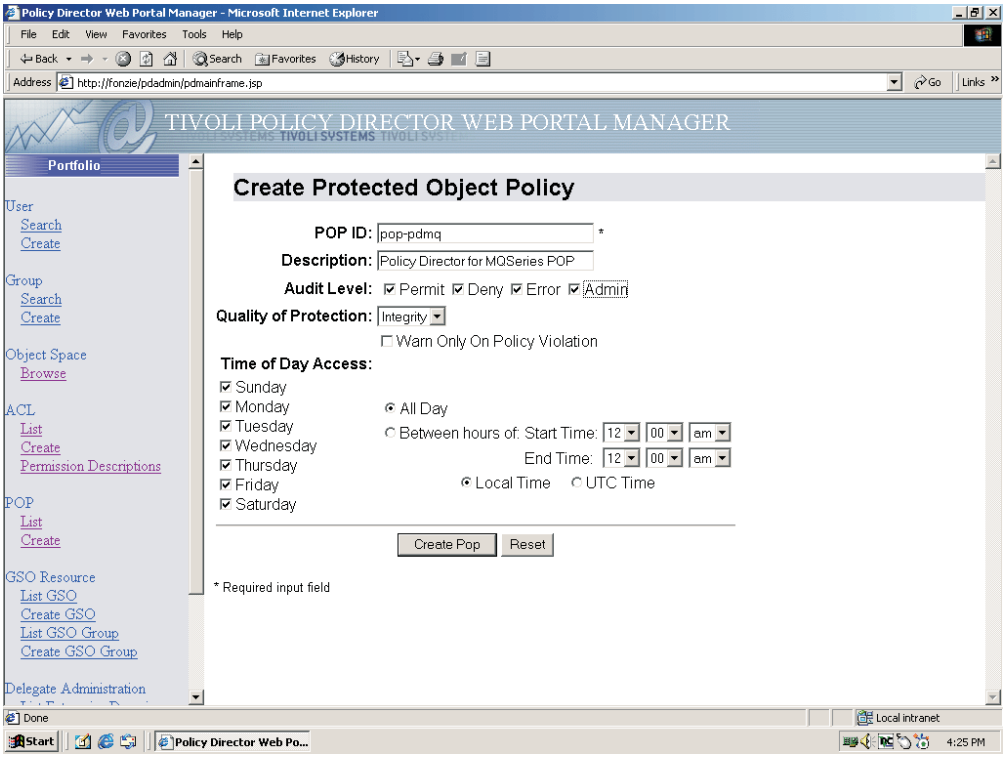

Figure 70. Create Protected Object Policy screen

3. Enter the name of the POP that you need to configure in the **POP ID** field.
4. Check the box next to each audit level that you need to apply to the specified object.
5. From the drop-down list in the **Quality of Protection** field, select the appropriate level of protection that you need to apply to the specified object.
6. Click **Create Pop**. The Create Protected Object Policy screen is displayed. See Figure 71.

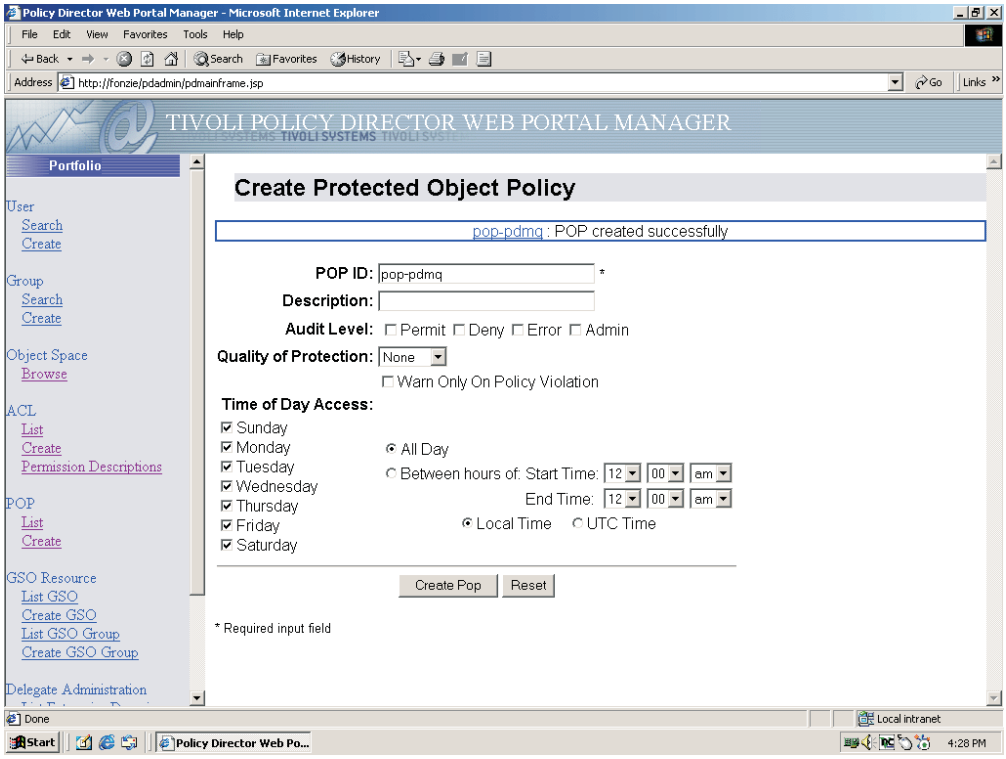

Figure 71. Verify that the POP is created

7. Review the following message that is displayed at the top of the screen: `pop-pdmq: POP created successfully`.
8. Under **POP** in the left-hand panel of the screen, click **List**. The POP List screen is displayed. See Figure 72 on page 117.

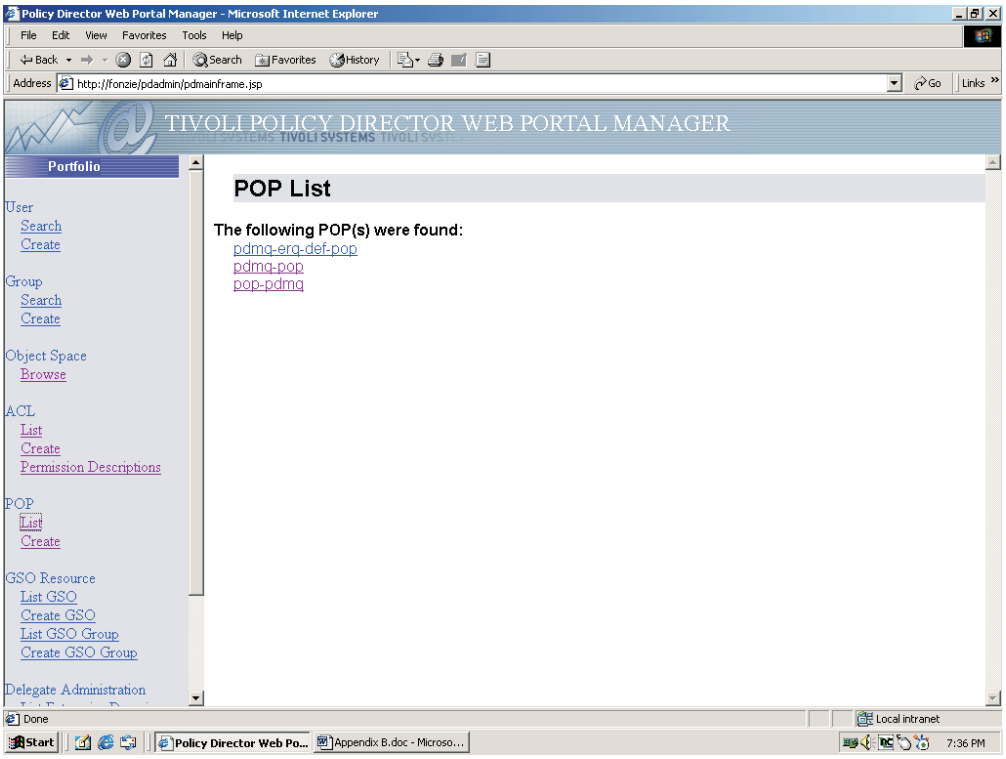

Figure 72. POP List screen

9. Click the name of the POP that you have configured. The POP Properties screen is displayed. See Figure 73.

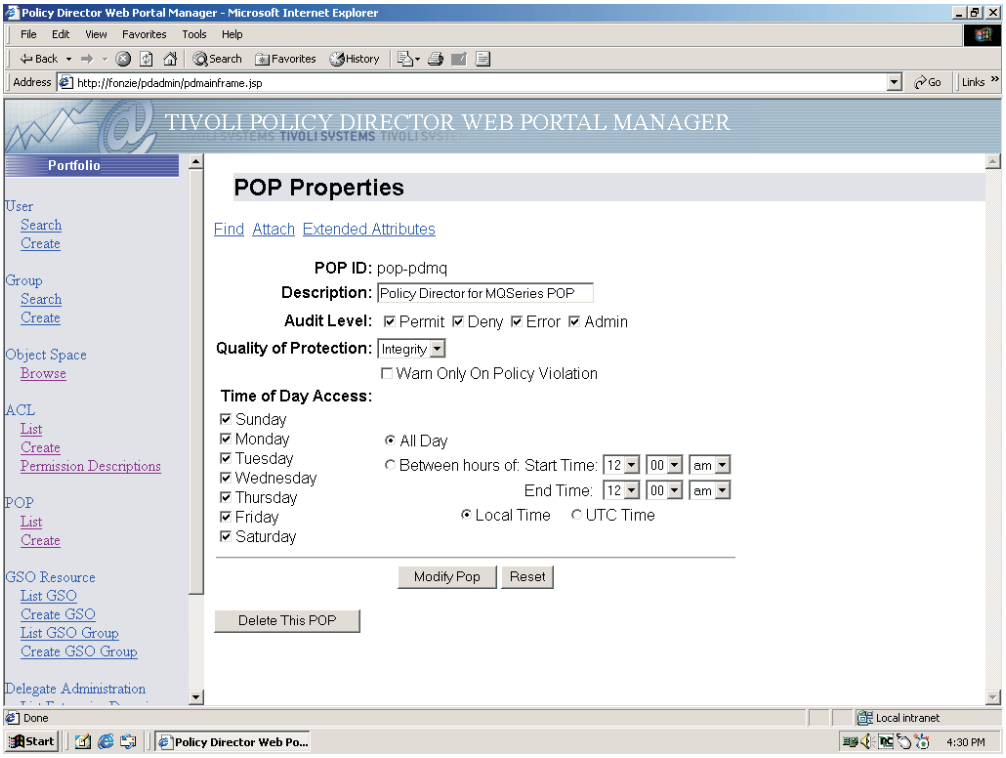

Figure 73. POP Properties screen

10. Click **Attach** to display the Attach POP screen. See Figure 74.

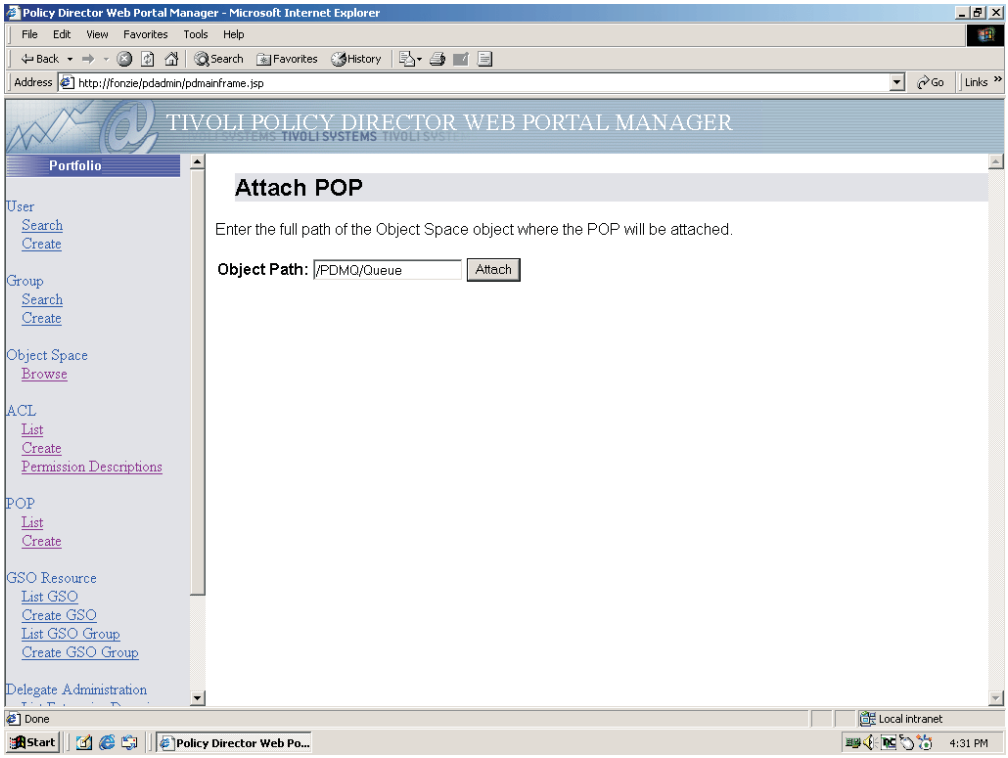

Figure 74. Attach POP screen

11. In the **Object Path** field, enter the path of the object to which you need to apply the POP that you have configured.
12. Click **Attach**.
13. Browse the object space to verify that the POP is attached. (See "Browsing the Tivoli Policy Director Object Space" on page 83 for instructions on how to browse the object space.) See Figure 75 on page 119.

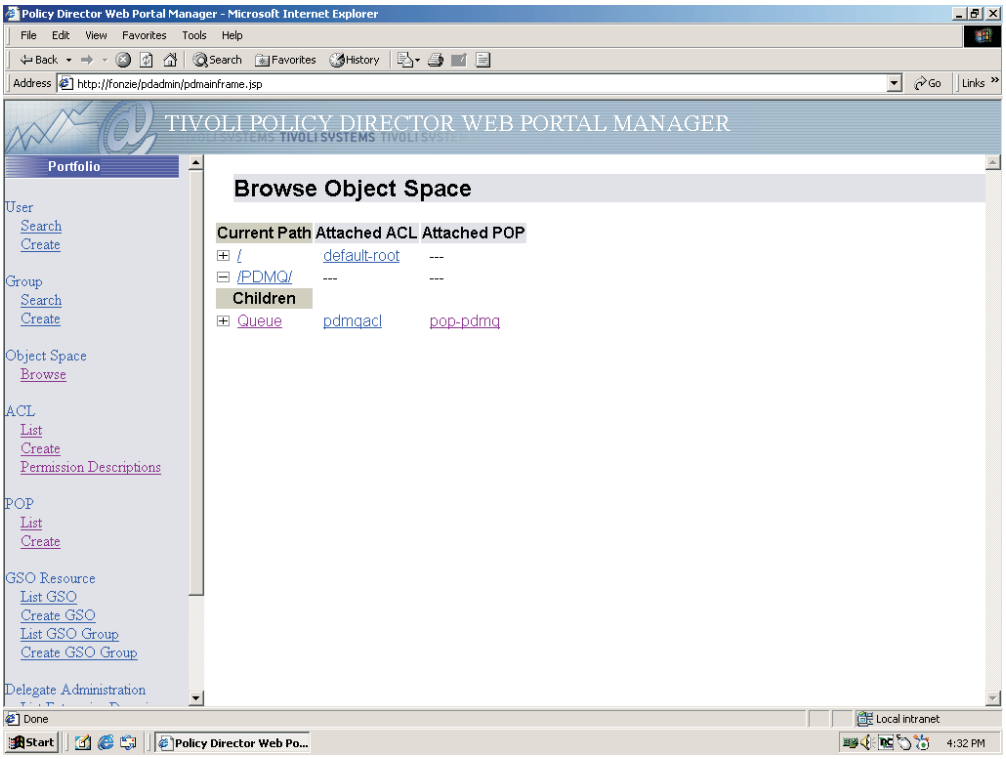

Figure 75. Verify that the POP is attached

Note: As shown in the example, you can apply the audit levels `Permit`, `Deny`, `Error`, and `Admin` and the `Integrity` level of protection to the POP `pop-pdmq`, and you can then attach the POP at the `/PDMQ/Queue` level of the object space.

Appendix B. Problem Determination

This appendix lists the messages that might be displayed. Numbered messages are of the form DRQ*AAnnnnS*:

DRQ Identifies a PDMQ message.

AA Identifies the component issuing the message.

nnnn Identifies the specific message number.

S Specifies the severity level:

The messages are in order by the 4-digit message number.

DRQDD0100E Policy Director for MQSeries cannot load configurations: reason code *error_code*

Explanation: The Policy Director for MQSeries initialization failed because the server could not obtain necessary information from the configuration file.

User Response: Make sure Policy Director for MQSeries is configured correctly.

DRQDD0101E Policy Director for MQSeries cannot initialize credential mapping: reason code *error_code*

Explanation: The Policy Director for MQSeries could not initialize its credential mapping subsystem because an internal error occurred.

User Response: Make sure Policy Director for MQSeries is configured correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0102E Policy Director for MQSeries cannot clean up credential mapping.

Explanation: The Policy Director for MQSeries could not de-initializa its credential mapping subsystem because an internal error occurred.

User Response: Make sure Policy Director for MQSeries is configured correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0103E Policy Director for MQSeries cannot clean up config resources.

Explanation: The Policy Director for MQSeries could not de-initialize its configuration resources because an internal error occurred.

User Response: Make sure Policy Director for MQSeries is configured correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0104E Policy Director for MQSeries internal error: Could not set process rlimit.

Explanation: The Policy Director for MQSeries Server could not set a process resource limit. This probably occurred because the local operating system identity under which the server was running did not have permission to change the limit.

User Response: Ensure that local operating system identity under which the Policy Director for MQSeries Server runs has permission to set system resource limits.

DRQDD0106E Policy Director for MQSeries Server internal error: Could not unlink file (*name nnnn***)**

Explanation: The Policy Director for MQSeries Server could not remove the indicated file.

User Response: Check that the Policy Director for MQSeries Server has write permission on the file.

DRQDD0107E Policy Director for MQSeries internal error: Could not start RPC listener

Explanation: The Policy Director for MQSeries Server could not bind to any port in the port range specified in the `pdmqazn.conf` file.

User Response: Ensure that the port range specified in `pdmqazn.conf file` is sufficiently large.

DRQDD0108E Policy Director for MQSeries Server is already running (PID *nnnn*)

Explanation: Only a single instance of the Policy Director for MQSeries Server may exist on a machine. You were attempting to start a second instance, which the server detected and disallowed.

User Response: Do not attempt to start the Policy Director for MQSeries Server when it is already running.

DRQDD0109E Could not create PID file (*file nnnn*)

Explanation: The Policy Director for MQSeries Server could not create the PDMQDaemon.pid file. It either does not have permission to do so, or does not have sufficient space in the filesystem containing the Policy Director for MQSeries log directory.

User Response: Check the Policy Director for MQSeries log directory (/var/pdmq/log on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path*\log on Windows platforms) to make sure PDMQDaemon.pid file, if it exists, is writable by the 'pdmq' user. Ensure that the filesystem containing the log directory has sufficient space available to hold the log file.

DRQDD0110E Could not create Port file (*file nnnn*)

Explanation: The Policy Director for MQSeries Server could not create the PDMQDaemon.port file. It either does not have permission to do so, or does not have sufficient space in the filesystem containing the /var/pdmq/log directory

User Response: Check the /var/pdmq/log directory to make sure PDMQDaemon.pid file, if it exists, is writable by the 'pdmq' user. If the filesystem lacks sufficient space, take measures to increase its available space.

DRQDD0111I Port file already exists (*file nnnn*)

Explanation: The Policy Director for MQSeries Server's PDMQDaemon.port file already exists with the desired port. Therefore, the server did not create new PDMQDaemon.port file.

User Response: No action is required.

DRQDD0112E Policy Director for MQSeries internal error: Could not become background process (*nnnn*).

Explanation: Either the pipe() or the fork() system call failed when the Policy Director for MQSeries server called it. The condition is usually caused by lack of memory or some other system resource.

User Response: Monitor the system resources consumed on the machine where the Policy Director for MQSeries is running. If necessary, increase the amount of memory or other required resources available to the server. Make sure that any MQSeries and Policy Director modifications to the operating system kernel parameters were made correctly.

DRQDD0113E Policy Director for MQSeries internal error: Could not start background process.

Explanation: Policy Director for MQSeries Server process ended abnormally before completing startup. This error usually occurs because the operating system is misconfigured, Policy Director for MQSeries is misconfigured, or the system is running low on memory.

User Response: Try to start the Policy Director for MQSeries Server again to ensure that the process was not killed accidently. Also, ensure that sufficient memory and other operating system resources are available to the server. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0114E Policy Director for MQSeries Server internal error: Could not create new thread *nnnn*.

Explanation: The Policy Director for MQSeries Server could not create or set attributes for a thread. This can occur if the system is running low on resources. Also, it can be caused by an internal error.

User Response: Ensure that sufficient memory and other operating system resources are available to the server. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0115E Invalid UNIX user name *name*.

Explanation: The local operating system user ID 'pdmq' did not exist

User Response: Ensure that the user 'pdmq' exists on the local operating system.

DRQDD0116E Invalid UNIX group name *name*.

Explanation: The local operating system group 'pdmq' did not exist

User Response: Ensure that the group 'pdmq' exists on the local operating system.

DRQDD0117E Could not change process GID (*GID*).

Explanation: The local operating system identity under which the Policy Director for MQSeries server was running did not have permission to change the group ID for the process.

User Response: Make sure that Policy Director for MQSeries server is started with root authority or as the privileged user 'pdmq'.

DRQDD0118E Could not change process UID (*UID*).

Explanation: The local operating system identity under which the Policy Director for MQSeries server was running did not have permission to change the user ID for the process.

User Response: Make sure that Policy Director for MQSeries server is started with root authority or as the privileged user 'pdmq'.

DRQDD0119E Policy Director for MQSeries cannot initialize Authorization sub-system

Explanation: Policy Director for MQSeries Server authorization subsystem initialization failed. The error log `pdmqd.log` gives details as to the possible causes.

User Response: Ensure that Policy Director is functioning correctly. Also, make sure that the filesystem containing the Policy Director for MQSeries var directory (/var/pdmq on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path* on Windows platforms) directory has sufficient space. For specific failure and recovery information, refer to previous error messages in the `pdmqd.log`.

DRQDD0120E Policy Director for MQSeries cannot initialize PKI mapping

Explanation: The Policy Director for MQSeries Server could not initialize its PKI mapping subsystem because an internal error occurred.

User Response: Ensure that Policy Director for MQSeries is configured correctly. If the mapping library (`pdmqmap.dll` on Windows; `pdmqmap.so` or `pdmqmap.a` on UNIX) is part of Policy Directory for MQSeries shipped package, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

If you are using a third-party mapping library, please contact the appropriate vendor for support.

DRQDD0121I Policy Director for MQSeries Server initialization succeeded

Explanation: This is an informational message that indicates the Policy Director for MQSeries Server has started

User Response: No action is required.

DRQDD0122E Policy Director for MQSeries server crashed-PID *nnnn*.

Explanation: The Policy Director for MQSeries server ended abnormally.

User Response: Restart the Policy Director for MQSeries server:

- UNIX: As root or the privileged user 'pdmqd', run **pdmqd -start**
- Windows: Restart from the Services panel.

DRQDD0125E Policy Director for MQSeries server shutdown failed - PID:*nnnn*.

Explanation: The Policy Director for MQSeries server cannot be terminated.

User Response: Make sure no process named **pdmqd** is running. If one exists, end the task manually.

DRQDD0126I Policy Director for MQSeries server shutdown succeeded.

Explanation: This is an informational message that Policy Director for MQSeries server has completed shutdown procedures.

User Response: No action is required.

DRQDD0127E Policy Director for MQSeries server cleanup failed - Aborting

Explanation: The Policy Director for MQSeries server could not shutdown cleanly. Preceding errors usually cause this problem.

User Response: Refer to the previous error messages in the `pdqmd.log` and the associated recovery actions.

DRQDD0128I Policy Director for MQSeries Server Process restarting ...*nnnn*

Explanation: This is an informational message that indicates the Policy Director for MQSeries Server process ended abnormally. The server monitor process automatically started a new server.

User Response: No action is required.

DRQDD0129I Becoming Monitor Process ...*nnnn*

Explanation: This is an informational message that indicates the Policy Director for MQSeries Server monitor process is watching the Policy Director for MQSeries Server for abnormal termination.

User Response: No action is required.

DRQDD0130E Policy Director for MQSeries Server initialization failed

Explanation: The Policy Director for MQSeries Server could not complete its initialization procedures.

User Response: Refer to the previous error messages in the `pdmqd.log` and the associated recovery actions. Ensure that Policy Director for MQSeries is configured correctly.

DRQDD0131E Invalid Port range specified in pdmqazn.conf file

Explanation: An incorrect port range was supplied for the Policy Director for MQSeries Server in the `pdmqazn.conf` file

User Response: Check the port range specified in `pdmqazn.conf` file. Make sure the syntax is correct and the range is large enough to ensure that enough free ports are available. Change the port range if no ports are available in the existing range.

DRQDD0134I Request Listener thread *nnnn* started

Explanation: This is an information message that indicates the Policy Director for MQSeries Server request listener thread started.

User Response: No action is required.

DRQDD0135I Policy Director for MQSeries Server listening on port *nnnn*.

Explanation: This is an information message that indicates the Policy Director for MQSeries Server request listener thread started.

User Response: No action is required.

DRQDD0200E Configuration type not found in Windows registry: *data*.

Explanation: A required Policy Director configuration value was not found in the Windows registry.

User Response: Make sure Policy Director is installed and configured correctly.

DRQDD0201E Policy Director for MQSeries configuration not initialized.

Explanation: The Policy Director for MQSeries server could not initialize its configuration subsystem because an internal error occurred.

User Response: Make sure Policy Director for MQSeries configured correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0202E Configuration entry not found.

Explanation: The Policy Director for MQSeries server could not find a required entry in the configuration subsystem.

User Response: Make sure that Policy Director and Policy Director for MQSeries configured correctly. Refer to the `pdmq.log` for specific details.

DRQDD0203E Could not open stanza file *stanza nnnn data*

Explanation: The Policy Director for MQSeries could not open a required configuration file.

User Response: Make sure Policy Director and Policy Director for MQSeries are installed and configured correctly.

DRQDD0204E Cannot determine Policy Director Version

Explanation: Cannot get Policy Director for MQSeries installation path from registry.

User Response: The Policy Director for MQSeries server could not get the Policy Director for MQSeries installation path from the Windows registry.

User Response: Make sure that Policy Director for MQSeries is configured correctly.

DRQDD0205E Cannot determine Policy Director Version

Explanation: Policy Director for MQSeries could not determine the Policy Director installation path.

User Response: Make sure that Policy Director is installed and configured correctly.

DRQDD0206E Cannot sniff configurations from config files.

Explanation: Policy Director for MQSeries could not obtain required configuration parameters from its configuration files.

User Response: Make sure that Policy Director and

Policy Director for MQSeries are configured correctly. Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0207E Policy Director for MQSeries cannot get pdmq information from the Authorization server.

Explanation: The Policy Director for MQSeries server could not get Policy Director for MQSeries information from the authorization server.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0208E Policy Director for MQSeries cannot get queue information from the Authorization server.

Explanation: The Policy Director for MQSeries server could not get queue information from the authorization server.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0209E Policy Director for MQSeries cannot get qmgr information from the Authorization server.

Explanation: The Policy Director for MQSeries server could not get queue manager information from the authorization server.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0210E Invalid map line in the mapping file.

Explanation: The Policy Director for MQSeries server detected a syntax error in the map.conf file.

User Response: Correct the error in the map.conf file.

DRQDD0211E Cannot get host name information

Explanation: The Policy Director for MQSeries server could not get hostname information from the system

User Response: Make sure that the local operating system host name resolution is working correctly. If problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0212E The Policy Director for MQSeries cannot update cached POP information.

Explanation: The Policy Director for MQSeries could not update its cache because an internal error occurred.

User Response: Ensure that Policy Director and Policy Director for MQSeries are configured and operating correctly. Refer to the previous error messages in the pdmqd.log and to associated recovery actions.

DRQDD0213E Policy Director for MQSeries internal error: Cannot allocate buffer.

Explanation: The Policy Director for MQSeries server could not allocate memory because the system is out of memory.

User Response: Increase the amount of memory available to the Policy Director for MQSeries server and retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0214E Encryption qop not found for qmgr *queue_manager* **queue** *queue***.**

Explanation: Policy Director for MQSeries could not get the encryption quality of protection for the queue.

User Response: Make sure that a POP exists for the queue in the Policy Director protected object space. Make sure Policy Director is operating correctly. Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0215E Signature qop not found for qmgr *queue_manager***, queue** *queue***.**

Explanation: The Policy Director for MQSeries could not get the signature quality of protection for the queue.

User Response: Make sure a POP exists for the queue in the Policy Director protected object space. Refer to the previous error messages in the pdmqd.log and to associated recovery actions.

DRQDD0216E Failed to get the error queue name for qmgr *queue_manager***.**

Explanation: Policy Director for MQSeries could not get the queue manager's error queue form the Policy Director protected object space. This problem usually occurs because an error queue is not defined in the protected object space.

User Response: Set the error-handling-Q extended attribute for the queue manager in the Policy Director protected object space. Make sure Policy Director is operating correctly.

DRQDD0217E No recipients found for queue manager *queue_manager***, queue** *queue***.**

Explanation: The Policy Director for MQSeries could not get the queue recipients list for the queue from the Policy Director protected object space. This problem usually occurs because the list of queue recipients is not defined in the protected object space.

User Response: Set the Q-recipients extended attribute for the queue in the Policy Director protected object space. Make sure Policy Director is operating correctly.

DRQDD0218E Mapping path not found.

Explanation: Policy Director for MQSeries could not open the `map.conf` file for reading.

User Response: Make sure the `map.conf` file exists in the Policy Director for MQSeries directory:

- /opt/pdmq/etc on UNIX platforms
- *Policy_Director_for_MQSeries_Installation_Path**etc on* on Windows platforms.

Make sure the file is readable by the 'pdmq' user.

DRQDD0219I Configuration item not found in Windows registry *data***.**

Explanation: Policy Director for MQSeries could not find an optional configuration parameter in the Windows registry.

User Response: No action is required unless the optional configuration parameter is necessary for correct Policy Director for MQSeries operation.

DRQDD0220E *parameter* **not found in stanza** *stanza* **in file** *file_name***.**

Explanation: Policy Director for MQSeries could not find a required configuration parameter in the indicated configuration file.

User Response: Make sure Policy Director is installed and configured correctly.

DRQDD0221I *parameter* **not found in stanza** *stanza* **in file** *file_name***.**

Explanation: Policy Director for MQSeries could not find an optional configuration parameter in the indicated configuration file.

User Response: No action is required unless the optional configuration parameter is necessary for correct operation.

DRQDD0222E Dump file could not be opened for writing. *file_name***.**

Explanation: The Policy Director for MQSeries could not open the configuration dump file for writing.

User Response: Make sure the 'pdmq' user can write to the dump file in the Policy Director for MQSeries directory:

- /var/pdmq/log on UNIX platforms
- *Policy_Director_for_MQSeries_Installation_Path*\log on Windows platforms.

Make sure that there is enough space in the filesystem for the dump file.

DRQDD0223E Unknown or unsupported user registry

Explanation: The Policy Director for MQSeries could not use the specified user registry.

User Response: Ensure that the Policy Director for MQSeries user registry is LDAP. Policy Director for MQSeries supports only LDAP.

DRQDD0300E Policy Director for MQSeries internal error: Invalid component Id *nnnn*

Explanation: A Policy Director for MQSeries Server internal command function was passed an invalid component ID.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information

DRQDD0301E Policy Director for MQSeries internal error: Invalid command Id *nnnn*

Explanation: A Policy Director for MQSeries Server internal command function was passed an invalid command ID.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information

DRQDD0302E Policy Director for MQSeries internal error: Memory corruption is detected. exiting ...

Explanation: The Policy Director for MQSeries Server detected that its memory was corrupted by an internal IPC message that was larger than available buffer space.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information

DRQDD0303E Policy Director for MQSeries Server internal error

Explanation: The Policy Director for MQSeries Server detected an internal error.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information

DRQDD0305E Could not create config dump

Explanation: The Policy Director for MQSeries server could not create a configuration dump. This problem usually occurs because the path does not exist, the server does not have permission to write to the file, or the filesystem containing the file does not have sufficient space.

User Response: Make sure that:

- The directory for the dump file exists
- The local operating system identity under which the Policy Director for MQSeries server is running has permission to write to the dump file
- Sufficient space exists in the filesystem into which the dump file is written.

Refer to the previous error messages in the `pdmqd.log` and the associated recovery actions.

DRQDD0307E PKI Mapping Update call failed

Explanation: The Policy Director for MQSeries Server could not update its PKI mapping cache.

User Response: Refer to the previous error messages in the pdmqd.log and the associated recovery actions.

DRQDD0403E No AZN configuration file found.

Explanation: The Policy Director for MQSeries server could not find the AZN configuration file.

User Response: Make sure that Policy Director for MQSeries is configured correctly.

DRQDD0404E Cannot create attribute list for initialization, Reason *error_code error_code***.**

Explanation: The Policy Director for MQSeries server received the indicated error code from the authorization subsystem when it attempted to create an attribute list.

User Response: Make sure that Policy Director for MQSeries is configured correctly.

DRQDD0405E Add attribute *attribute* **failed.** *error_code error_code*

Explanation: The Policy Director for MQSeries server received the indicated error code from the authorization subsystem when it attempted to add an attribute.

User Response: Make sure that Policy Director for MQSeries is configured correctly.

DRQDD0406E SSL key stash file *file_name* **has been truncated.**

Explanation: The Policy Director for MQSeries server SSL key stash file has been truncated.

User Response: Make sure the server SSL key stash file has not been tampered with and is readable by the local operating system user 'pdmq'.

DRQDD0407E SSL key stash file *file_name* **is inaccessible.**

Explanation: The Policy Director for MQSeries server could not access its SSL key stash file.

User Response: Make sure the server SSL key stash file is readable by user 'pdmq'.

DRQDD0408E Authorization subsystem initialization failed, reason *error_code error_code***.**

Explanation: The Policy Director for MQSeries server could not initialize the authorization subsystem. This error usually indicates that the Policy Director for MQSeries user registry is not LDAP.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured correctly.

DRQDD0409E DCE server principal is null

Explanation: The Policy Director for MQSeries server detected a NULL DCE server principal. This error usually indicates that the Policy Director for MQSeries user registry is not LDAP.

User Response: Make sure that Policy Director for MQSeries user registry is LDAP. Policy Director for MQSeries supports only LDAP as the user registry. If the problem occurs when using LDAP as the user registry, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0410E DCE server key is null

Explanation: The Policy Director for MQSeries server detected a NULL DCE server key. This error usually indicates that the Policy Director for MQSeries user registry is not LDAP.

User Response: Make sure that Policy Director for MQSeries user registry is LDAP. Policy Director for MQSeries supports only LDAP as the user registry. If the problem occurs when using LDAP as the user registry, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0411E DCE server keytab login failed: principal *principal* **keytab** *keytab* **error** *error_code*:*error_code*.

Explanation: The Policy Director for MQSeries server DCE keytab login failed. This error usually indicates that the Policy Director for MQSeries user registry is not LDAP.

User Response: Make sure that Policy Director for MQSeries user registry is LDAP. Policy Director for MQSeries supports only LDAP as the user registry. If the problem occurs when using LDAP as the user registry, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0412I Authorization subsystem initialization succeeded.

Explanation: The Policy Director for MQSeries server successfully initialized the authorization subsystem.

User Response: No action is required.

DRQDD0414E Authorization subsystem shutdown error: *error_code error_code*

Explanation: The Policy Director for MQSeries server encountered an error during authorization system shutdown.

User Response: If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0415I Authorization subsystem shutdown succeeded.

Explanation: The Policy Director for MQSeries server successfully shutdown the authorization system.

User Response: No action is required.

DRQDD0417E Cannot get name list from perminfo, AZN error code *error_code error_code*

Explanation: The Policy Director for MQSeries server could not get the name list from the perminfo internal function because it received the indicated error code from the authorization subsystem.

User Response: Make sure that Policy Director for MQSeries is configured correctly.

DRQDD0420E Queue manager name is missing.

Explanation: The Policy Director for MQSeries server did not supply a queue manager name to an internal authorization call.

User Response: If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0421E Cannot build any AZN credentials, Policy Director for MQSeries error code *error_code*

Explanation: The Policy Director for MQSeries server could not create credentials because the indicated internal error occurred.

User Response: Refer to the previous error messages in the `pdmq.log` and the associated recovery actions.

DRQDD0422E Authorization check failed, error code *error_code error_code*.

Explanation: A Policy Director for MQSeries server authorization check failed.

User Response: Make sure the user attempting to access the object has sufficient permission. Make sure that Policy Director is operating correctly.

DRQDD0423I Authorization check result *data*.

Explanation: The Policy Director for MQSeries server authorization received the indicated result from an authorization check.

User Response: No action is required.

DRQDD0424E Authorization check return value unknown.

Explanation: The Policy Director for MQSeries server received an unknown result from an authorization check.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0425E PD audit initialization failed.

Explanation: The Policy Director for MQSeries server could not initialize the audit subsystem because an internal error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0426I PD audit initialization succeeded.

Explanation: The Policy Director for MQSeries server successfully initialized the audit subsystem.

User Response: No action is required.

DRQDD0427E Attach audit sink failed. PD error code *error_code***.**

Explanation: The Policy Director for MQSeries server could not attach the audit sink because the indicated Policy Director error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0428E Add audit event failed, PD error code *error_code***.**

Explanation: The Policy Director for MQSeries server could not add an audit event to the audit trail because the indicated Policy Director error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0429E Add audit text for *data* **failed. PD error code** *error_code***.**

Explanation: The Policy Director for MQSeries server could not add audit text to an event because the indicated Policy Director error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0430E Commit audit event failed, PD error code *error_code***.**

Explanation: The Policy Director for MQSeries server could not commit an audit event because the indicated Policy Director error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDD0431E Start audit event failed, PD error code *error_code***.**

Explanation: The Policy Director for MQSeries server could not start an audit event because the indicated Policy Director error occurred.

User Response: Make sure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1000E Policy Director for MQSeries internal error: Could not allocate memory.

Explanation: The Policy Director for MQSeries Interceptor could not allocate memory because a system resource problem occurred.

User Response: Make sure the system has sufficient resources to run the application. Then restart the application.

DRQDM1001E Policy Director for MQSeries internal error: could not allocate memory.

DRQDM1002E Policy Director for MQSeries internal error: GSKIT could not allocate *data***. GSKIT reason code** *error_code***.**

Explanation: A Policy Director for MQSeries Interceptor GSKit call failed with the indicated reason code because the system could not allocate some resource.

User Response: Make sure the system has sufficient resources to run the application. Consult the Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages for the explanation of the GSKit reason code and take corrective action. Then restart the application.

DRQDM1003W Policy Director for MQSeries internal error: GSKIT could not release *data***. GSKIT reason code** *error_code***.**

Explanation: A Policy Director for MQSeries Interceptor GSKit call failed with the indicated reason code because the system could not some resource back to the system.

User Response: Consult the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1004E Policy Director for MQSeries internal error:

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1005E Policy Director for MQSeries internal error: Memory corruption.

Explanation: The Policy Director for MQSeries Interceptor failed because it detected that an internal call corrupted memory.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1100I Initializing Policy Director for MQSeries interceptor.

Explanation: The server has started.

DRQDM1101I Policy Director for MQSeries Build *nnnn nnnn*

Explanation: This is an informational message indicating that the server has started.

DRQDM1102I Policy Director for MQSeries interceptor initialization succeeded.

Explanation: This is an informational message indicating that the server has started.

DRQDM1103E Policy Director for MQSeries interactive login initialization failed.

Explanation: Policy Director for MQseries Interactive Login initialization failed because an internal error occurred.

User Response: Make sure the Java runtime is installed and the CLASSPATH is set correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1104E Policy Director for MQSeries server port file could not be opened.

Explanation: Policy Director for MQSeries daemon port file does not exist or has incorrect permissions.

User Response: Make sure the Policy Director for MQSeries server is running. On UNIX platforms, the process 'pdmqd' should exist; on Windows NT and 2000, the Services panel should indicate that the Policy Director of MQSeries Service is running. If the server is not running, restart it and and try the operation again. If the server cannot be started, check the error log to determine the cause of the failure and take corrective action.

DRQDM1105E Policy Director for MQSeries server port *port* **is invalid.**

Explanation: Policy Director for MQSeries daemon request port is invalid.

User Response: Ensure that the TCP port range specified in the `pdmqazn.conf` file is correct and restart the Policy Director for MQSeries Server.

DRQDM1200E Policy Director for MQSeries interceptor could not connect to queue manager. MQSeries compcode *nnnn* : reason *nnnn*

Explanation: The Policy Director for MQSeries Interceptor could not connect to the MQSeries queue manager.

User Response: Make sure the specified queue manager is running. Note the MQSeries completion code (CompCode) and reason code (Reason), and check the MQSeries documentation for an explanation of the codes.

DRQDM1201E Default queue manager name could not be found.

Explanation: The Policy Director for MQSeries Interceptor could not determine the default queue manager. Most likely, no default queue manager was set.

User Response: Make sure the default queue manager is set in the `mqs.ini`file on UNIX platforms, or in the Windows registry on Windows platforms.

DRQDM1202E Queue manager group feature is not supported.

Explanation: Queue manager group is not supported.

User Response: Check for supported features.

DRQDM1203E Policy Director for MQSeries internal error. Queue manager information could not be saved.(*error_code*)

Explanation: The Policy Director for MQSeries Interceptor could not save the queue manager connection (hconn) information because an internal error occurred.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1204W Policy Director for MQSeries internal error. Queue manager information could not be deleted.(*error_code*)

Explanation: The Policy Director for MQSeries Interceptor could not delete the queue manager handle because an internal error occurred.

User Response: If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1300E Policy Director for MQSeries internal error. GSS credential could not be released. GSKIT reason code (*error_code*).

Explanation: GSS status code.

DRQDM1301E Policy Director for MQSeries could not find current queue manager name. *error_code*.

Explanation: Policy Director for MQSeries status code.

DRQDM1302E Policy Director for MQSeries internal error: Queue information could not be saved. *error_code*.

Explanation: The Policy Director for MQSeries Interceptor could not save queue information because an internal error occurred.

User Response: Make sure the system has sufficient resources for Policy Director for MQSeries. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html for more information

---

DRQDM1303E Policy Director for MQSeries could not resolve current queue name from object handle. *error_code*.

Explanation: Policy Director for MQSeries status code.

DRQDM1304I Resolved queue manager name is *name*, resolved queue name is *name*.

Explanation: Display the resolved queue manager name and queue name.

DRQDM1305E Policy Director for MQSeries interceptor could not get current process information. *error_code*.

Explanation: UNIX platform only.

DRQDM1306E Policy Director for MQSeries interceptor could not find current PKI information. *error_code*.

Explanation: Policy Director for MQSeries status code.

DRQDM1307E Policy Director for MQSeries could not acquire public key credential. *error_code*.

Explanation: Policy Director for MQSeries status code.

DRQDM1308E Policy Director for MQSeries could not map user public key credential to Policy Director userid. *error_code*.

Explanation: Policy Director for MQSeries status code.

DRQDM1309E Policy Director for MQSeries authorization check failed. *error_code*.

Explanation: A Policy Director for MQSeries Interceptor call to the authorization subsystem failed with the indicated error.

User Response: Consult the Policy Director Administration Guide for information about the error code. Ensure that Policy Director and Policy Director for MQSeries are configured and operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1310E MQOPEN() call failed: MQSeries reason code (*nnnn***).** *error_code*.

Explanation: MQ reason code.

DRQDM1311E Policy Director for MQSeries internal error: IDUP environment could not be established: GSKit reason code (0x%x).

Explanation: The Policy Director for MQSeries Interceptor call to setup the GSKit IDUP environment failed with the indicated reason code.

User Response: Consult the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1312E Policy Director for MQSeries internal error: Dynamic queue information could not be added.

Explanation: The Policy Director for MQSeries Interceptor could not add dynamic queue information because an internal error occurred.

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1313I MQOPEN() call succeeded.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully called the MQSeries MQOPEN() function.

User Response: No action is required.

DRQDM1314E Policy Director for MQSeries internal error: Queue information could not be deleted. *error_code*.

Explanation: The Policy Director for MQSeries Interceptor could not delete dynamic queue information because an internal error occurred.

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1315E Policy Director for MQSeries internal error. IDUP environment could not be abolished. GSKIT reason code (*error_code***).**

Explanation: The Policy Director for MQSeries Interceptor could not release the GSKit IDUP environment because an internal error occurred.

User Response: Consult the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action.

If the problem occurs persistently, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1316E MQCLOSE() call failed, MQSeries reason code (*nnnn***).**

Explanation: The Policy Director for MQSeries Interceptor call to the MQSeries MQCLOSE() function failed with the indicated MQSeries reason code.

User Response: Consult the MQSeries documentation for an explanation of the error code and suggested corrective action.

Ensure that the queue manager is operational, the queue exists, and the Policy Director for MQSeries entries in the Policy Director protected object space are present and correct.

Make sure that Policy Director and Policy Director for MQSeries are configured and running correctly.

To obtain more information about the problem, turn on

the NOTICE serviceability level in the Policy Director routing file.

DRQDM1317W Policy Director for MQSeries internal error. Dynamic queue information could not be deleted. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not delete dynamic queue information because an internal error occurred.

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1318E Unauthenticated user could not perform required message protection.

Explanation: The Policy Director for MQSeries Interceptor detected that the user attempting to sign or encrypt a message does not have a public key identity.

User Response: Make sure the current user has an entry in the map.conf file. Also, make sure the kdb file specified in `map.conf` file is readable by the current user.

DRQDM1400E Policy Director for MQSeries internal error: Queue information could not be resolved from current queue object handle. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not resolve queue information from current queue object handle because the current queue was not opened via Policy Director for MQSeries, or it was closed.

User Response: Ensure that the queue has not already been opened by another MQSeries application and that it has not been previously closed. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1401I Policy Director for MQSeries interceptor bypassed the MQPUT() call for system queue *queue*.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor detected and bypassed an MQPUT() for a SYSTEM queue. Policy Director for MQSeries it does not intercept MQPUT() calls onto SYSTEM queues.

User Response: No action is required.

DRQDM1402E Policy Director for MQSeries internal error: could not find local queue manager's CodedCharSetId.*error_code*.

Explanation: The Policy Director for MQSeries Interceptor could not find the local queue manager's CodedCharSetId because an internal error occurred.

User Response: Ensure that the queue manager is connected via Policy Director for MQSeries. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1403E Quality of protection *nnnn* for queue *queue* is invalid.

Explanation: The Policy Director for MQSeries Interceptor detected that the quality of protection (QoP) specified in the Policy Director protected object space for the queue is invalid. Only the values 'privacy', 'integrity', or 'none' may be specified.

User Response: Ensure that the Protected Object Policy (POP) attached to the queue object in the Policy Director protected object space has a valid QoP value.

DRQDM1404E Policy Director for MQSeries internal error: could not get recipient information for queue *queue*. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not get the list of queue recipients from the Policy Director protected object space. This usually occurs because the Q-recipients extended attribute is not attached to the queue obje

User Response: Ensure that a Q-recipients extended attribute containing the recipient list is defined in the Policy Director protected object space for the queue.

DRQDM1405E There is no recipient information defined for queue *queue* which requires privacy protection.

Explanation: The Policy Director for MQSeries Interceptor got the Q-recipients attribute attached to the queue object in the Policy Director protected object space. However, no recipients were defined in the attribute.

User Response: Ensure that the Q-recipients extended attribute defined in the Policy Director protected object space for the queue contains at least one recipient.

DRQDM1406E Policy Director for MQSeries internal error: Recipient's DN could not be found in GSKIT key database. GSKIT return code (*error_code*) reason code (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not find a recipient distinguished name (DN) in the GSKit key database. This usually occurs because the local key database does not contain the DN, or is corrupt.

User Response: Use the GSKit Ikeyman tool to verify that the key database is intact and contains the recipient DN. Consult the GSKIT information in the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action.

DRQDM1407E Policy Director for MQSeries internal error: GSKIT API call gss_import_name() failed. GSKIT return code *error_code*, reason *error_code*.

Explanation: The Policy Director for MQSeries Interceptor received the indicated error from the GSKit gss_import_name() function.

User Response: Use the GSKit Ikeyman tool to verify that the key database is intact and contains the recipient DN. Consult the GSKIT information in the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1408E Policy Director for MQSeries internal error: Recipient's name could not be added to recipient's name set. GSKIT return code (*error_code*) reason code (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not add a recipient name to a recipient name set because an internal error occurred. This problem can occur if the system is running low on resources. .

User Response: Ensure that the system has sufficient resources and retry the operation. Also, consult the GSKit information in the *Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages* for the explanation of the GSKit reason code and take corrective action.

If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1409E Quality of protection *error_code* for queue *queue* is invalid.

Explanation: The Policy Director for MQSeries Interceptor detected that the quality of protection (QoP) specified in the Policy Director protected object space for the queue is invalid. Only the values 'privacy', 'integrity', or 'none' may be specified.

User Response: Ensure that the Protected Object Policy (POP) attached to the queue object in the Policy Director protected object space has a valid QoP value.

DRQDM1410E Policy Director for MQSeries internal error: Message could not be protected: GSKIT return code (*error_code*) reason (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not sign or encrypt a message because the indicated GSKit error occurred. This can happen for several reasons, all of which are internal failures:

- The GSKit encryption strength is invalid
- The Policy Director credential is expired
- The message recipient information is incorrect

User Response: Consult the GSKit information in the Tivoli Policy Director for MQSeries Version 3.8 Guide to Global Security Toolkit (GSKIT) Messages for the explanation of the GSKit reason code and take corrective action.

If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1411E Policy Director for MQSeries internal error: MQPUT() event audit failed. *error_code*.

Explanation: The Policy Director for MQSeries Interceptor could not audit an MQPUT() event because an internal error occurred

User Response: Make sure the filesystem that contains the audit log has space sufficient for the entire log. Also, ensure that the system has sufficient resources available to run Policy Director for MQSeries. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1412I Message with no protection has been put into queue *queue*.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully put a message with a protection level of 'none' onto the selected queue.

User Response: No action is required.

DRQDM1413I Message with integrity protection has been put into queue *queue*.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully put a message with a protection level of 'integrity' onto the selected queue.

User Response: No action is required.

DRQDM1414I Message with privacy protection has been put into queue *queue*.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully put a message with a protection level of 'privacy' onto the selected queue.

User Response: No action is required.

DRQDM1500E Policy Director for MQSeries internal error: Queue information could not be resolved from current queue object handle. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not resolve queue information from current queue object handle because the current queue was not opened via Policy Director for MQSeries, or it was closed.

User Response: Ensure that the queue has not already been opened by another MQSeries application and that it has not been previously closed. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1501I Policy Director for MQSeries found a valid 'PDMQ' format header from the current message.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor found a valid Policy Director for MQSeries header in the current message.

User Response: No action is required.

DRQDM1502E Policy Director for MQSeries interceptor could not get a 'PDMQ' format header from the current message: MQ reason (*nnn*).

Explanation: The Policy Director for MQSeries Interceptor could not get the current message because an MQSeries error occurred.

User Response: Consult the MQSeries documentation for an explanation of the reason code and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1503E The 'PDMQ' format header from the current message is invalid.

Explanation: The Policy Director for MQSeries Interceptor did not find a valid Policy Director for MQSeries header in the current message.

User Response: Check the Quality of Protection (QoP) setting for the queue object in the Policy Director protected object space. If the QoP setting for the queue is not 'none', ensure that no MQSeries plain text messages are routed to this queue.

DRQDM1504E Policy Director for MQSeries interceptor could not find current queue manager's CodedCharSetId. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not find the current queue manager either because it does not exist, or is connected via internal MQSeries APIs.

User Response: Specify a queue that exists. If the queue exists, ensure that no other MQSeries application is connected to it. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1505E Policy Director for MQSeries internal error. Dynamic queue information could not be added. (*error_code*).

Explanation: The Policy Director for MQSeries Interceptor could not add dynamic queue information, possibly because an error occurred with a system resource, the MQSeries queue manager, or Policy Director.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries. Also, ensure that the MQSeries queue manager and Policy Director are operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1506E Message quality of protection *nnnn* **does not match quality of protection** *nnnn* **set for queue** *queue_name***.**

Explanation: The Policy Director for MQSeries Interceptor detected a quality of protection (QoP) mismatch between the queue and a message in the queue. QoP mismatches usually occur because the queue QoP is changed between the message put and get.

User Response: Ensure that the same queue quality of protection is used for the message put and get. When the queue QoP is changed, clean up all messages in the queue before resuming normal operation.

DRQDM1507E Policy Director for MQSeries internal error. MQGET() event audit failed. (*error_code***).**

Explanation: The Policy Director for MQSeries Interceptor could not audit an MQPUT() event because an internal error occurred. .

User Response: Make sure the filesystem that contains the audit log has space sufficient for the entire log. Also, ensure that the system has sufficient resources available to run Policy Director for MQSeries. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1508I Message was signed by *name* **at** *time***.**

Explanation: This is an informational message used to indicate that a message was signed, and to display the signer's name and message timestamp.

User Response: No action is required.

DRQDM1509I Message was signed and encrypted by *name* **at** *time***.**

Explanation: This is an informational message used to indicate that a message was signed and encrypted, and to display the signer's name and message timestamp.

User Response: No action is required.

DRQDM1510I Message was not protected.

Explanation: This is an informational message used to indicate that a message was neither signed nor encrypted.

User Response: No action is required.

DRQDM1511E Message does not have a valid protection type.

Explanation: The Policy Director for MQSeries Interceptor detected an invalid protection type in a message header. This usually occurs because the MQSeries message header is corrupt.

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1512E Message encryption strength *strength1* **is weaker than required strength** *strength2***.**

Explanation: The Policy Director for MQSeries Interceptor detected that a message did not meet they encryption strength required by the queue. This usually occurs when encryption strength for a queue is upgraded while there were still messages in the queue.

User Response: Ensure that the same encryption strength is used for the message put and get. When the queue encryption strength is changed, clean up all messages in the queue before resuming normal operation.

DRQDM1513E Policy Director for MQSeries could not map a message sender *string* **to a Policy Director user. (***error_code***)**

Explanation: The Policy Director for MQSeries Interceptor could not map the indicated Distinguished Name (DN) contained in a message to a Policy Director principal.

User Response: Ensure that the DN is mapped in the map.conf file. Also, ensure that Policy Director is operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1514E Policy Director for MQSeries interceptor could not obtain the authorization information from the Policy Director server. (*error_code***).**

Explanation: The Policy Director for MQSeries Interceptor could not obtain authorization information from Policy Director.

User Response: Make sure that Policy Director is operating correctly.

DRQDM1515W The message sender *name* does not have the permission to put the message on queue *queue_name*.

Explanation: This is a warning message that indicates the user represented by the Distinguished Name (DN) does not have permission to put a message on the queue.

User Response: No action is normally required. However, if the problem occurs frequently, ensure that the user is not trying to attack the queue.

DRQDM1516E Policy Director for MQSeries interceptor could not get the whole message: MQSeries reason code *nnnn*.

Explanation: The Policy Director for MQSeries Interceptor could not get a whole message because the indicated MQSeries error occurred. Though this condition only happens rarely, it can occur when Policy Director for MQSeries and some other MQSeries application are both getting messages from the same queue.

User Response: Use care when mixing MQSeries applications with Policy Director for MQSeries. Consult the MQSeries documentation for more information about the reason code and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1517E There is no error handling queue defined for the current queue manager.

Explanation: There is no error handling queue defined for the current queue manager. Policy Director for MQSeries interceptor will try to use SYSTEM.DEFAULT.DEAD.LETTER.QUEUE.

User Response: Ensure that each queue manager protected by Policy Director for MQSeries has an error queue defined for it.

DRQDM1518I Policy Director for MQSeries interceptor has put a defective message in the error handling queue. *nnnn*

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor put a message it could not interpret on the specified error handling queue.

User Response: Make sure all senders put valid messages onto queues protected by Policy Director for MQSeries.

DRQDM1519E Policy Director for MQSeries interceptor failed to put defective message into error handling queue. MQ reason code (*nnnn*)

Explanation: The Policy Director for MQSeries Interceptor was unable to put a message it could not interpret onto the error handling queue because the indicated MQSeries error occurred.

User Response: Consult the MQSeries documentation for more information about the reason code. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1520E Policy Director for MQSeries internal error: Message could not be unprotected: GSKit error code 0xd0000, reason 0x2.

Explanation: The Policy Director for MQSeries server detected a syntax error in the map.conf file.

User Response: Correct the syntax error in the map.conf file.

DRQDM1617E Policy Director for MQSeries internal error. Stanza file *name* could not be locked: *nnnn nnnn*.

Explanation: The Policy Director for MQSeries Interceptor could not lock the indicated configuration file.

User Response: Ensure that the local operating system identity under which the program is running has permission to lock the configuration file.

DRQDM1618E Policy Director for MQSeries internal error. Stanza file *name* could not be opened: *nnn nnn*.

Explanation: The Policy Director for MQSeries Interceptor could not open the indicated configuration file for reading.

User Response: Ensure that local operating system identity under which the program is running has permission to open the configuration file for reading.

DRQDM1619E Policy Director for MQSeries internal error. Unexpected NULL argument in *nnn* routine.

Explanation: The indicated Policy Director for MQSeries Interceptor routine could not find a required value in a configuration file.

User Response: Inspect the configuration file and correct any omissions.

DRQDM1620E Policy Director for MQSeries internal error. Required configuration item *stanza key* is missing.

Explanation: The Policy Director for MQSeries Interceptor could not find a required value in a configuration file. The offending item denoted by the substitution variables is of the form [stanza:key].

User Response: Inspect the configuration file and correct any omissions.

DRQDM1621E Policy Director for MQSeries internal error. The *nnn*.* audit level specified in the *file_name* configuration file is invalid.

Explanation: The Policy Director for MQSeries Interceptor detected an invalid audit level in the indicated configuration file.

User Response: Inspect the configuration file and correct the audit level specification.

DRQDM1622E Policy Director for MQSeries internal error. The audit level string specified in the *file_name* configuration file is badly formatted.

Explanation: The Policy Director for MQSeries Interceptor detected an invalid audit level string in the indicated configuration file.

User Response: Inspect the configuration file and correct the audit level string.

DRQDM1623E Policy Director for MQSeries installation path could not be found in the Windows registry.

Explanation: The Policy Director for MQSeries Interceptor could not find the Policy Director for MQSeries entries in the Windows registry.

User Response: Ensure that Policy Director for MQSeries is installed and configured correctly.

DRQDM1624E Policy Director for MQSeries could not find PKI user profile for OS user *user*.

Explanation: The Policy Director for MQSeries Interceptor could not find a public key infrastructure (PKI) user profile for the indicated local operating system identity. This usually occurs because no entry for the user exists in the `map.conf` file.

User Response: Ensure that the `map.conf file` contains a mapping entry for the user.

DRQDM1625E Policy Director for MQSeries could not find PKI user key for OS user *user*.

Explanation: The Policy Director for MQSeries Interceptor could not find the public key public key infrastructure (PKI) private key for the indicated local operating system user mapped in the map.conf file.

User Response: Ensure that the public key database (kdb) file specified in map.conf file for the local operating system user is accessible by the Policy Director for MQSeries Interceptor.

DRQDM1803E Policy Director for MQSeries internal error: AZN credential could not be built. AZN error code. *error_code*:*error_code*

Explanation: The Policy Director for MQSeries Interceptor could not build a Policy Director credential because an authorization subsystem error occurred.

User Response: Ensure that Policy Director is operating correctly and that Policy Director for MQSeries is configured correctly. Consult the *Policy Director Administration Guide* for an explanation of the error code. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1804E Policy Director for MQSeries internal error: Attribute names from *data* could not be found. AZN error code *error_code*:*error_code*.

Explanation: The Policy Director for MQSeries Interceptor could not find an extended attribute name attached to the indicated object in the Policy Director protected object space.

User Response: Ensure that the object's attribute is correctly set in the Policy Director protected object space. Also, ensure that Policy Director is operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1807E Policy Director for MQSeries internal error: Value *value* for attribute *attribute* in object *object* is invalid.

Explanation: The Policy Director for MQSeries Interceptor could not find the extended attribute value attached to the indicated object in the Policy Director protected object space.

User Response: Ensure that the object's attribute is correctly set in the Policy Director protected object space. Also, ensure that Policy Director is operating

correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1808W Policy Director for MQSeries internal error: Attribute *attribute* **has** *nnn* **values. Only one is expected.**

Explanation: The Policy Director for MQSeries Interceptor detected more than one value defined for the indicated extended attribute in the Policy Director protected object space, and used the last value specified.

User Response: Ensure that only one value is specified for the attribute in the Policy Director protected object space.

DRQDM1811E Policy Director for MQSeries internal error: AZN credential could not be obtained. AZN error code. *error_code* **:***error_code*

Explanation: The Policy Director for MQSeries Interceptor could not obtain a credential from Policy Director because an authorization subsystem error occurred.

User Response: Ensure that Policy Director is operating correctly and that Policy Director for MQSeries is configured correctly. Consult the Policy Director Administration Guide form an explanation of the error code. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1814E Policy Director for MQSeries internal error: Unauthenticated user could not query *data***s information. (***error_code***)**

Explanation: The Policy Director for MQSeries Interceptor detected that an unauthenticated user was attempting to query the indicated information and disallowed the operation.

User Response: Ensure that no unauthorized users are attempting to query Policy Director for MQSeries information. If the user should be authorized, make sure the user is correctly defined in Policy Director and Policy Director for MQSeries.

DRQDM1815E Policy Director for MQSeries internal error: AZN cache could not be initialized. *error_code*

Explanation: The Policy Director for MQSeries Interceptor could not initialize the authorization database cache.

User Response: Ensure that Policy Director is operating correctly. Also, ensure that sufficient space exists in the Policy Director for MQSeries database directory (/var/pdmq/db on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path*\db on Windows platforms). If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1816E Policy Director for MQSeries internal error: Socket could not be created. OS error *nnn*

Explanation: Ensure that the system has sufficient resources to run Policy Director for MQSeries. Consult the local operating system documentation for an explanation of the error and suggested corrective actions. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

User Response: The Policy Director for MQSeries Interceptor could not create a socket because the indicated local operating system error occurred.

DRQDM1817E Policy Director for MQSeries internal error: Socket could not be bound to port. *nnnn***.OS error** *nnn***.**

Explanation: The Policy Director for MQSeries Interceptor could not bind a socket to the port because the indicated local operating system error occurred.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries. Consult the local operating system documentation for an explanation of the error and suggested corrective actions. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html
for more information.

DRQDM1819E Policy Director for MQSeries interceptor could not connect to the server. *error_code*

Explanation: The Policy Director for MQSeries Interceptor could not connect to the Policy Director for MQSeries Server.

User Response: Ensure that the Policy Director for MQSeries Server is operating correctly. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1820I The Policy Director for MQSeries interceptor connected to the server.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully connected to the Policy Director for MQSeries Server.

User Response: No action is required.

DRQDM1821I The Policy Director for MQSeries interceptor disconnected from the server.

Explanation: This is an informational message that indicates the Policy Director for MQSeries Interceptor successfully disconnected from the Policy Director for MQSeries Server.

User Response: No action is required.

DRQDM1824E Policy Director for MQSeries internal error: Socket could not be closed. OS error (*nnnn*).

Explanation: The Policy Director for MQSeries Interceptor could not close a socket because the indicated local operating system error occurred.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries. Consult the local operating system documentation for an explanation of the error and suggested corrective actions. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1827E Policy Director for MQSeries internal error: Message buffer could not be received correctly.

Explanation: The Policy Director for MQSeries Interceptor could not receive an interprocess communication (IPC) message from the Policy Director for MQSeries Server because the message size did not match the message buffer size.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries and retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1828E Policy Director for MQSeries internal error: Message buffer could not be sent correctly.

Explanation: The Policy Director for MQSeries Interceptor could not send an interprocess communication (IPC) message from the Policy Director for MQSeries Server because the message size did not match the message buffer size.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries and retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDM1829E Policy Director for MQSeries internal error: Failed to complete a call to the server.

Explanation: A Policy Director for MQSeries Interceptor call to the Policy Director for MQSeries Server did not complete. This problem usually occurs because the server is not running.

User Response: Ensure that the the Policy Director for MQSeries Server is running correctly and retry the operation. Examine the error log to determine if a server failure has occurred and take corrective action. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDT2002W Policy Director Run Time environment is not configured. Check your Policy Director configuration.

Explanation: Policy Director Run Time environment is not configured.

User Response: Policy Director Run Time environment should be configured before configuring Policy Director for MQSeries.

DRQDT2003E ERROR: Unsupported operating system type:

Explanation: Policy Director for MQSeries does not support this Operating system.

User Response: Policy Director for MQSeries does not support this Operating system.

DRQDT2004W WARNING: this script must be executed by 'root' (uid = 0).

Explanation: This program must be executed by root.

User Response: Make sure user is running this command as root.

DRQDT2005E Invalid option found in the command line *name*.

Explanation: An invalid command line option was passed in.

User Response: Rerun the command specifying the right options.

DRQDT2006W Policy Director for MQSeries is already configured.

Explanation: Policy Director for MQSeries is already configured.

User Response: Do not attempt to configure Policy Director for MQSeries when it is already configured.

DRQDT2007E Required option *option* not found in the command line.

Explanation: A required option was not specified on the command line.

User Response: Rerun the command specifying the correct set of options.

DRQDT2008E pdmqazn.conf.template file not found in Policy Director for MQSeries install path.

Explanation: Policy Director for MQSeries installation creates a template file named pdmqazn.conf.template in the Policy Director for MQSeries etc directory (/opt/pdmq/etc on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path*\etc on Windows platforms). The configuration program could not find this template file. The most likely cause is that Policy Director for MQSeries is not installed properly.

User Response: Reinstall Policy Director for MQSeries and retry the operation.

DRQDT2009E MQSeries processes are active. Please close all MQSeries processes and then retry operation.

Explanation: MQSeries processes are active. Please close all MQSeries processes and then retry operation.

User Response: All MQSeries activity should be stopped before running this command.

DRQDT2011E Error in configuring /PDMQ/Queue in the object space. Refer to the configuration log for more information.

Explanation: Configuration failed to configure initial /PDMQ/Queue objects in the object space. Refer to the configuration log for more information.

User Response: Ensure that Policy Director is running correctly. Also, refer to the previous error messages in config.log and take appropriate corrective action.

DRQDT2013E Error in unconfiguring /PDMQ/Queue in the object space. Refer to the configuration log for more information.

Explanation: The configuration program could not unconfigure the Policy Director for MQSeries base objects (/PDMQ/Queue) in the Policy Director protected object space.

User Response: Ensure that Policy Director is running correctly. Also, refer to the previous error messages in config.log and take appropriate corrective action.

DRQDT2015E Error in configuring SSL connection to Policy Director server. Please refer to configuration log for more information.

Explanation: Policy Director for MQSeries Secure Sockets Layer (SSL) configuration failed.

User Response: Ensure that Policy Director is running correctly. Also, refer to the *Policy Director Administration Guide* for more information about this error.

DRQDT2017E Policy Director for MQSeries SSL unconfiguration failed. Please refer to configuration log for more information.

Explanation: The configuration program could not unconfigure the Policy Director for MQSeries Server Secure Sockets Layer (SSL).

User Response: Ensure that Policy Director is running correctly. Also, refer to the *Policy Director Administration Guide* for more information about this error.

DRQDT2018E Error starting Queue Manager *name*.

Explanation: The Policy Director for MQSeries configuration program could not start the queue manager because an MQSeries error occurred.

User Response: Refer to MQSeries error logs for more information. Then consult the MQSeries documentation for an explanation and take corrective action.

DRQDT2019E Error starting Command server for Queue Manager *name*.

Explanation: The Policy Director for MQSeries configuration program could not start the MQSeries command server for the indicated queue manager because an MQSeries error occurred.

User Response: Refer to MQSeries error logs for more information. Then consult the MQSeries documentation for an explanation and take corrective action.

DRQDT2021E Error in importing MQSeries Queues to Policy Director Object space. Please refer to configuration log for more information.

Explanation: The Policy Director for MQSeries configuration program could not create queue information in the Policy Director protected object space. The **mq2pd** program failed.

User Response: Ensure that Policy Director is running correctly. Also, refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2024E Error in removing queues from the object space. Please refer to configuration log for more information.

Explanation: The Policy Director for MQSeries configuration program could not remove queue objects from the Policy Director protected object space.

User Response: Ensure that Policy Director is running correctly. Also, refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2026E Error in enabling Policy Director for MQSeries interceptor. Please refer to configuration log for more information.

Explanation: The configuration program could not enable the Policy Director for MQSeries Interceptor.

User Response: Ensure that Policy Director is running correctly. Also, refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2028E Error in disabling Policy Director for MQSeries Interceptor. Please refer to configuration log for more information.

Explanation: The configuration program could not disable the Policy Director for MQSeries Interceptor.

User Response: Refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2029E Policy Director for MQSeries Interceptor is not enabled. Please use -enable option to enable it.

Explanation: The configuration program was attempting a Policy Director for MQSeries operation while the Policy Director for MQSeries Interceptor was not enabled.

User Response: Use **-enable** option to enable the Policy Director for MQSeries Interceptor.

DRQDT2034W Policy Director for MQSeries daemon is running. Please stop the daemon and retry operation.

Explanation: The Policy Director for MQSeries Server was running when the configuration program attempted an operation that required the server to be down.

User Response: Stop the Policy Director for MQSeries Server.

DRQDT2035W Policy Director for MQSeries interceptor is already enabled.

Explanation: The configuration program attempted to enable the Policy Director for MQSeries Interceptor when it was already enabled.

User Response: Do not attempt to enable the Policy Director for MQSeries Interceptor when it is enabled.

DRQDT2036W Policy Director for MQSeries interceptor is already disabled.

Explanation: The configuration program attempted to disable the Policy Director for MQSeries Interceptor when it was already disabled.

User Response: Do not attempt to disable the Policy Director for MQSeries Interceptor when it is disabled.

DRQDT2089W Check if Policy Director for MQSeries is installed.

Explanation: The configuration program detected that MQSeries is not installed correctly.

User Response: Make sure Policy Director for MQSeries is installed properly and retry the operation.

DRQWT2090W Check if MQSeries is installed.

Explanation: The configuration program detected that MQSeries is not installed correctly.

User Response: Make sure MQSeries is installed properly and retry the operation.

DRQWT2091W Check if Policy Director for MQSeries is installed.

Explanation: The configuration program detected that Policy Director for MQSeries is not installed correctly.

User Response: Make sure Policy Director for MQSeries is installed properly and retry the operation.

DRQWT2092E Policy Director for MQSeries Internal error : Error writing configuration state.

Explanation: The configuration program cannot write the Policy Director for MQSeries configuration state to the PDMQRTEConfigured file (`/opt/pdmq/.configure/PDMQRTEConfigured` on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path* `\.configure\PDMQRTEConfigured` on Windows platforms).

User Response: Ensure that the local operating system identity under which the configuration program is running has read permission on the file. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQWT2093E Error reading current Policy Director for MQSeries configuration state.

Explanation: The configuration program cannot read the Policy Director from MQSeries configuration state to the PDMQRTEConfigured file (`/opt/pdmq/.configure/PDMQRTEConfigured` on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path*`\.configure\PDMQRTEConfigured` on Windows platforms).

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQWT2097E Policy Director for MQSeries Internal error : could not allocate memory.

Explanation: The Policy Director for MQSeries configuration program could not allocate memory.

User Response: Ensure that the system has sufficient resources to run Policy Director for MQSeries.

DRQWT2098E One of the required text field marked by (*) is not specified.

Explanation: The Policy Director for MQSeries configuration program detected missing required text fields and marked each with an asterisk (*) in the Policy Director Configuration GUI.

User Response: Specify all the required text fields.

DRQWT2101W Some MQ Process is using MQM.dll. Please close all MQ processes using this dll.

Explanation: Policy Director for MQSeries interceptor cannot be enabled if any process is using MQSeries mqm.dll shared library.

User Response: Make sure all MQSeries processes are stopped and then retry configuration.

DRQWT2102E Configuration tried stopping all MQSeries activity but failed to do so. Please exit configuration, close all MQProcesses and retry configuration.

Explanation: Configuration tried stopping all MQSeries activity but failed to do so. Please exit configuration, close all MQProcesses and retry configuration.

User Response: Please exit configuration, close all MQProcesses and retry configuration.

DRQWT2103W Queue Manager *manager* **does not exist in MQSeries.**

Explanation: The specified MQSeries queue manager must exist before the configuration program can add the queue manager representation to the Policy Director protected object space.

User Response: Specify a valid MQSeries queue manager.

DRQWT2104W Queue Manager *manager* **already exists in MQSeries.**

Explanation: The configuration program could not add the specified queue manager representation to the Policy Director protected object space.

User Response: Use the **update** configuration option to update the queue manager.

DRQWT2105E ERROR: adding entry *entry* **to** *file***.**

Explanation: The configuration program could not add an entry to the configuration file `pdmqazn.conf`.

User Response: Ensure that the pdmqazn.conf file (`/opt/pdmq/etc/pdmqazn.conf` on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path* `\etc\pdmqazn.conf` on Windows platforms) exists and the local operating system identity under which the program is running has write permission on the file.

DRQWT2106E ERROR: copy of *file* **to** *file* **failed.**

Explanation: The configuration program could not copy pdmqazn.conf.template file to `pdmqazn.conf`.

User Response: Make sure Policy Director for

MQSeries is installed properly, the pdmqazn.conf.template file (/opt/etc/pdmqazn.conf.template on UNIX platforms, *Policy_Director_for_MQSeries_Installation_Path* \etc\pdmqazn.conf.template) exists, and the local operating system identity under which the program is running has read permission on the file.

DRQWT2107E ERROR: copy operation of Policy Director pdpthread.dll to Policy Director for MQSeries pthread.dll failed.

Explanation: The configuration program could not copy the Policy Director pdpthread.dll to pthread.dll.

User Response: Make sure the local operating system identity under which the program is running has permission to read *Policy_Director_Installation_Path*\lib\pdpthread.dll and has permission to write *Policy_Director_for_MQSeries_Installation_Path*\lib\pthread.dll.

DRQWT2108E ERROR: getting LDAP host from registry.

Explanation: The configuration program could not obtain the LDAP hostname from Windows registry.

User Response: Make sure the Policy Director Run Time Environment is configured and that Policy Director is operating correctly.

DRQWT2109E ERROR: ivconf/pdconf does not exist.

Explanation: The configuration program could not find pdconf.exe under the Policy Director installation path.

User Response: Make sure Policy Director is installed and configured properly.

DRQWT2110E ERROR: getting LDAP port from registry.

Explanation: The configuration program could not get the LDAP port number from the Windows registry.

User Response: Make sure the Policy Director Run Time Environment is configured correctly.

DRQWT2113E Policy Director for MQSeries Internal error : MQSeries registry query failed to get the names of Queue managers on current machine.

Explanation: The configuration program could not query the MQSeries queue manager names from the Windows registry under the key

```
\HKEY_LOCAL_MACHINE
\SOFTWARE
\IBM
\MQSeries
\CurrentVersion
\Status
\QueueManager
```

User Response: Ensure that MQSeries is installed properly.

DRQWT2114E Policy Director for MQSeries Internal error : MQSeries registry query failed to get the names of Queue managers on current machine.

Explanation: The configuration program could not query the MQSeries queue manager names from the Windows registry under the key

```
\HKEY_LOCAL_MACHINE
\SOFTWARE
\IBM
\MQSeries
\CurrentVersion
\Status
\QueueManager
```

User Response: Ensure that MQSeries is installed properly.

DRQWT2115E ERROR: Policy Director for MQSeries Configuration tried starting the command server but failed to do so.

Explanation: The configuration program could not start the MQSeries command server because an MQSeries error occurred.

User Response: Please refer to the MQSeries error logs for failures. Then consult the MQSeries documentation for an explanation of the error and suggested corrective action.

DRQWT2116E ERROR: Policy Director for MQSeries Configuration tried stopping the Queue manager but failed to do so.

Explanation: The configuration program could not stop the MQSeries queue manager because an MQSeries error occurred.

User Response: Please refer to the MQSeries error logs for failures. Then consult the MQSeries documentation for an explanation of the error and suggested corrective action.

DRQDT2201I Policy Director for MQSeries is not enabled.

Explanation: The configuration program detected an attempt to disable the Policy Director for MQSeries Interceptor when the interceptor was already disabled.

User Response: Do not attempt to disable the interceptor when it is already disabled.

DRQDT2204E Warning MQSeries is running. Please stop all MQSeries processes before doing this operation.

Explanation: The configuration program could not perform the operation because an MQSeries process was active.

User Response: Close all MQSeries processes and retry operation.

DRQDT2205E Check whether MQ shared library *library* exists and whether user has write permission on the file.

Explanation: The MQSeries shared library could not be accessed.

User Response: Make sure that MQSeries is installed and configured properly.

DRQDT2207E Policy Director for MQSeries libraries do not exist. Please install Policy Director for MQSeries before configuring.

Explanation: The configuration program has detected that Policy Director for MQSeries is not installed correctly.

User Response: Ensure that Policy Director for MQSeries is installed correctly.

DRQDT2210E Check the file permissions on MQSeries library *library*.

Explanation: The configuration program could not successfully run the UNIX **chmod** command against the indicated file. This error can happen if the local operating system identity under which the program is running is not 'root.'

User Response: Make sure the local operating system identity running the program is 'root.'

DRQDT2211E MQSeries library *library* does not exist. Check whether MQSeries is properly installed.

Explanation: The configuration program could not find the MQSeries shared library.

User Response: Check whether MQSeries is installed correctly.

DRQDT2212E MQSeries Original library cannot be copied.

Explanation: The configuration program could not save a copy of the original MQSeries shared library.

On Windows platforms, the program could not copy *MQSeries_Installation_Path*`\bin\mqm.dll` to *MQSeries_Installation_Path*`\bin\origmqm.dll`

On the Solaris platform, the program could not copy `/opt/mqm/lib/libmqm.so` to `/opt/mqm/lib/liborigmqm.so`

On the AIX platform, the program could not copy `/usr/lpp/mqm/lib/libmqm_r.a` to `/usr/lpp/mqm/lib/liborigmqm_r.a`

User Response: Ensure that the local operating system identity under which the program is running has permission to copy the original shared library. Also, ensure that sufficient space exists in the filesystem that contains the MQSeries shared library.

DRQDT2214E Policy Director for MQSeries library cannot be copied. Some MQ process is using MQSeries shared library. Please close all MQSeries processes and try this operation again.

Explanation: The configuration program cannot enable or disable the Policy Director for MQSeries Interceptor while MQSeries was running because it could not copy the Policy Director for MQSeries shared library.

User Response: Make sure no MQSeries processes are running when trying to enable or disable the Policy Director for MQSeries Interceptor. Also, ensure that the local operating system identity under which the program is running has permission to copy the file.

DRQDT2217E Configuration failed to set Policy Director for MQSeries interceptor as enabled.

Explanation: The configuration program could not enable the Policy Director for MQSeries Interceptor because previous errors prevented it from succeeding.

User Response: Refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2219E Configuration failed to set Policy Director for MQSeries interceptor as disabled.

Explanation: The configuration program could not disable the Policy Director for MQSeries Interceptor because previous errors prevented it from succeeding.

User Response: Refer to the previous error messages in `config.log` and take appropriate corrective action.

DRQDT2220E Policy Director for MQSeries is not installed.

Explanation: The configuration program detected that Policy Director for MQSeries is not installed.

User Response: Make sure Policy Director for MQSeries is installed properly and then retry the operation.

DRQDT2221E Cannot find Policy Director for MQSeries and/or MQSeries installation

Explanation: The configuration program could not find the Policy Director for MQSeries and/or the MQSeries installation.

User Response: Ensure that Policy Director for MQSeries and MQSeries are installed correctly.

DRQDT2222E Cannot get MQSeries install path from registry path *path*.

Explanation: The configuration program has determined that MQSeries is not installed properly because it cannot find the MQSeries installation path in the Windows registry.

User Response: Ensure that MQSeries is installed correctly.

DRQDT2223E Copy of *file* to *file* failed.

Explanation: The configuration program could not save a copy of the original MQSeries shared library.

On Windows platforms, the program could not copy *MQSeries_Installation_Path*`\bin\mqm.dll` to *MQSeries_Installation_Path*`\bin\origmqm.dll`

On the Solaris platform, the program could not copy `/opt/mqm/lib/libmqm.so` to `/opt/mqm/lib/liborigmqm.so`

On the AIX platform, the program could not copy `/usr/lpp/mqm/lib/libmqm_r.a` to `/usr/lpp/mqm/lib/liborigmqm_r.a`

Failed to copy MQSeries mqm.dll to MQSeries origmqm.dll

User Response: Ensure that the local operating system identity under which the program is running has permission to copy the original shared library. Also, ensure that sufficient space exists in the filesystem that contains the MQSeries shared library.

DRQDT2224E Copy of *file* to *file* failed. Some MQ Process is using *file*. Please close all MQ Processes and try enabling interceptor again.

Explanation: The configuration program could not enable the Policy Director for MQSeries Interceptor because an MQSeries process was active.

User Response: Close all MQSeries processes and retry operation.

DRQDT2225E Copy of *file* to *file* failed. Some MQ Process is using *file*. Please close all MQ Processes and try enabling interceptor again.

Explanation: The configuration program could not enable the Policy Director for MQSeries Interceptor because an MQSeries process was active.

User Response: Close all MQSeries processes and retry operation.

DRQDT2322E Connect Queue Manager failed: Completion code = *nnnn*: Reason = *nnnn*.

Explanation: The configuration program could not contact the queue manager. Either the queue manager is not running, or the configuration program could not start the queue manager.

User Response: Start the queue manager and retry the operation. Consult the MQSeries documentation for and explanation of the reason code and suggested corrective action.

DRQDT2323E Create Namespace failed: Completion code = *nnnn*: Reason = *nnnn*.

Explanation: The configuration program could not query the queue information from MQSeries.

User Response: Make sure the MQSeries command server is running for the queue manager.

DRQDT2324E MQSeries Command Server is not running!!

Explanation: The configuration program detected that the MQSeries command server was not running for the queue manager. The operation requires the command server to be running.

User Response: Start the MQSeries command server and retry the operation.

DRQDT2326E Error creating error queue *queue* for Queue Manager *queue_manager*.

Explanation: The configuration program could not create the indicated MQSeries error handing queue.

User Response: Ensure that the MQSeries queue manager is running. Attempt to create the error handling queue via MQSeries commands.

DRQDT2327I Error queue *queue* already exists in Queue Manager *queue_manager*.

Explanation: The configuration program detected that the indicated MQSeries error handling queue already existed and, therefore, did not attempt to create a new error handling queue.

User Response: No action is required.

DRQDT2400E Policy Director login failed.

Explanation: Policy Director login failed because the configuration program provided an incorrect admin/password.

User Response: Ensure that the Policy Director administrator name and password are correct. Also, ensure that Policy Director is running correctly.

DRQDT2401E Failed to get Policy Director install path from Windows registry.

Explanation: The program could not get the Policy Director installation path from the Windows registry.

User Response: Ensure that Policy Director is installed properly.

DRQWT2404E Error setting interactive login to ON.

Explanation: The configuration program could not set interactive login to the ON state.

User Response: Ensure that Policy Director for MQSeries is installed properly.

DRQWT2405E Error setting interactive login to OFF.

Explanation: The configuration program could not set interactive login to the OFF state.

User Response: Ensure that Policy Director for MQSeries is installed properly.

DRQWT2407E One instance of Policy Director for MQSeries Login process is already running.

Explanation: The Policy Director for MQSeries Login process detected that another login process is already running.

User Response: Run only a single instance of the Policy Director for MQSeries Login process at a time.

DRQWT2408E Configuration failed to get GSKIT install path from Windows registry.

Explanation: The configuration program could not get the GSKIT path from the Windows registry.

User Response: Make sure GSKIT is installed correctly.

DRQWT2409E Configuration failed to get Policy Director for MQSeries install path from Windows registry.

Explanation: The configuration program could not get the Policy Director for MQSeries installation path from the Windows registry.

User Response: Ensure that Policy Director for MQSeries is installed properly.

DRQDT2412E Configuration failed to get information from Policy Director configuration files.

Explanation: The configuration program could not get information from Policy Director configuration files.

User Response: Make sure Policy Director Run Time Environment (PDRTE) is configured correctly.

DRQDT2414E Configuration cannot validate the information it got from configuration files.

Explanation: The configuration program could not validate the information it got from configuration files.

User Response: Make sure the Policy Director Run Time Environment (PDRTE) is configured correctly.

DRQDT2415W Configuration cannot open file *file* **for reading.**

Explanation: Policy Director for MQSeries could not open the indicated Policy Director configuration file.

User Response: Make sure Policy Director Run Time Environment (PDRTE) is configured and that the local operating system identity under which the program is running has read permissions on the file.

DRQDT2416I Configuration cannot find entry *entry* **in stanza** *stanza* **of file** *file***.**

Explanation: The configuration program could not find the specified entry in the indicated configuration file.

User Response: Make sure Policy Director Run Time Environment (PDRTE) is configured and that the local operating system identity under which the program is running has read permissions on the file.

DRQDT2425E Configuration failed to find some required parameters from Policy Director configuration files.

Explanation: The configuration program could not find some required parameters from Policy Director configuration files.

User Response: Make sure Policy Director Run Time

Environment (PDRTE) is configured correctly.

DRQDT2426E Configuration failed to get SSL Keyfile or password information from the configuration files.

Explanation: The configuration program could not get the Secure Sockets Layer (SSL) keyfile or password information from the configuration file.

User Response: Make sure the Policy Director Run Time Environment (PDRTE) is configured correctly.

DRQDT2433E MQSeries Queues cannot be enumerated. Check that Queue Manager and Command server are running.

Explanation: The configuration program could not configure the name space because an MQSeries error occurred.

User Response: Ensure that the MQSeries queue manager and command server are running.

DRQDT2436E Configuration cannot get the list of queue managers from Policy Director server.

Explanation: The configuration program could not query the list of queue managers in the Policy Director protected object space.

User Response: Make sure that Policy Director is running correctly.

DRQDT2437E Error getting the list of queues from Policy Director for queue manager *manager***.**

Explanation: The configuration program could not query the list of queues in the Policy Director protected object space for the indicated queue manager.

User Response: Make sure that Policy Director is running correctly.

DRQDT2500W Policy Director for MQSeries is not configured. Please use -all option to configure.

Explanation: The configuration program detected that it had not performed base Policy Director for MQSeries configuration.

User Response: Use **-all** option to configure Policy Director for MQSeries.

DRQDT2501E Policy Director for MQSeries service cannot be started.

Explanation: The program cannot start the Policy Director for MQSeries Server.

User Response: Ensure that Policy Director for MQSeries is configured correctly. Also, examine the `pdmqd.log` for other errors and take appropriate corrective action.

DRQDT2502E Policy Director for MQSeries service cannot be stopped.

Explanation: The program cannot disable the Policy Director for MQSeries Server.

User Response: Ensure that Policy Director for MQSeries is configured correctly. Also, examine the `pdmqd.log` for other errors and take appropriate corrective action.

DRQDT2508W Queue Manager *manager* **has not been added to the object space. You must use -add option to add queues for this Queue Manager.**

Explanation: The configuration program detected and disallowed an attempt to update the indicated queue manager because the queue manager had not been added to the Policy Director protected object space.

User Response: Use the **-add** option to add the queue manager to Policy Director protected object space.

DRQDT2509E Queue Manager *name* **has already been added to the object space. You must use -update option to update queues for this Queue Manager.**

Explanation: The configuration program detected and disallowed an attempt to add the indicated queue manager because the queue manager already existed in the Policy Director protected object space.

User Response: Use the **-update** option to update queues for the current queue manager in the Policy Director protected object space.

DRQDT2510E The user must have administrative privileges on their local host to perform this operation.

Explanation: The local operating system identity under which the program was running did not have the required administrative privileges.

User Response: Retry the operation using a local operating system identity that has administrative privileges.

DRQDH3000E Message was put on the error handling queue because of its quality of protection does not match the queue quality of protection.

Explanation: Quality of protection mismatches usually occur because the queue quality of protection was

changed between the message put and get.

User Response: Make sure the same queue quality of protection is used for the message get and put.

DRQDH3001E Message was put on the error handling queue because it could not be unprotected or verified.

Explanation: Depending on the queue quality of protection, the message either could not be decrypted or its signature could not be verified. This can happen for several reasons:

- The recipient has the wrong private key to decrypt or verify the message.
- The sender has the wrong public key certificate for the recipient.
- The message was altered in transmission.

User Response: Make sure that the recipient has the correct private key, the sender has the correct public key certificate for the recipient, and no message tampering, intentional or otherwise, has occurred.

DRQDH3002E Unexpected error. Error message reason code *error_code* **not defined.**

Explanation: The reason code that the error handling utility got from the message is undefined.

User Response: Retry the operation. If the problem persists, contact your IBM service representative. Refer to

http://techsupport.services.ibm.com/guides/handbook.html

for more information.

DRQDH3019E File *file_name* **could not be opened to store the message number** *number***.**

Explanation: The usual reasons a file cannot be opened are lack of directory, permission, or filesystem space.

User Response: Make sure the directory exists, that you have permission to write in it, and that there is enough space in the filesystem.

DRQDH3020E File name was not specified. The message number *number* **will only be displayed on the screen.**

Explanation: A file name must be specified to store the message.

User Response: Retry the operation specifying the file name.

Appendix C. Quick Start

This appendix provides a brief set of instructions for starting up Tivoli Policy Director for MQSeries, Version 3.8.1. You can use these instructions to try out the product or to test your system to make sure that Tivoli Policy Director for MQSeries is correctly installed. However, these instructions do not replace the installation instructions contained in Chapter 2, configuration instructions in Chapters 3 and 5, and mapping instructions in Chapter 6.

Using the Quick Start Files

This appendix contains sections on the following topics:

- "Setting Up a Queue Manager and Queues"
- "Installing and Configuring Tivoli Policy Director for MQSeries" on page 152
- "Creating a Tivoli Policy Director User" on page 153
- "Creating a Tivoli Policy Director Group" on page 153
- "Adding a User to a Group" on page 154
- "Creating a Tivoli Policy Director Protected Object Policy" on page 154
- "Creating the Tivoli Policy Director Access Control List" on page 154
- "Completing the Administration Process" on page 154

  Note: Completing the administration process includes attaching the protected object policy (POP), attaching the access control list (ACL), and setting the extended attributes.
- "Using a Self-Signed Certificate as a Public Key Infrastructure Identity" on page 155
- "Using a Pre-Made Mapping File" on page 155
- "Mapping Public Key Infrastructure Identities to Tivoli Policy Director Users" on page 55
- "Updating the Tivoli Policy Director for MQSeries Daemon" on page 156
- "Using the MQSeries Sample Application to Get and Put Messages from a Queue" on page 156

These instructions assume that IBM MQSeries is installed and a working Tivoli Policy Director server is available. Also, for Quick Start purposes, you must use the command files contained in the pdmqsamples.zip file in the /doc directory on the product CD. You need to unzip this file and use the extracted files in the sections where these files are being used. To begin using Tivoli Policy Director for MQSeries, perform the instructions in each of the following sections, in sequence.

Setting Up a Queue Manager and Queues

To start using Tivoli Policy Director for MQSeries, you must first create a queue manager and a queue using the following steps:

1. Create a queue manager named QM by entering the following command:

   ```
   crtmqm QM
   ```
2. Start MQSeries using your new queue manager by entering the following command:

   ```
   strmqm QM
   ```

3. Run the following command in MQSeries to create the queues Q.NONE, Q.INTEGRITY, and Q.PRIVACY:

   ```
   runmqsc QM <pdmqqdef.mqrc
   ```

   where:

   pdmqqdef.mqrc
   Specifies a command file from the pdmqsamples.zip file.

   If you want to create your own queues, you can use a text editor to modify the pdmqqdef.mqrc file and enter your own queue names.

4. To test your MQSeries configuration, run **amqsput** and **amqsget** to PUT and GET a message to each of the new queues. Make sure that your path includes the following directories: for AIX or Solaris paths, use *mq_install_path*/samp/bin; for Windows 2000 or Windows NT, use *mq_install_path*\bin.

   Enter the following commands; be sure to type a test message after each use of the **amqsput** command:

   ```
   amqsput Q.NONE QM
   amqsget Q.NONE QM
   amqsput Q.INTEGRITY QM
   amqsget Q.INTEGRITY QM
   amqsput Q.PRIVACY QM
   amqsget Q.PRIVACY QM
   ```

Installing and Configuring Tivoli Policy Director for MQSeries

After you have configured the MQSeries environment and verified that it is working, you can add security to this environment using Tivoli Policy Director for MQSeries. Perform the following steps:

1. To stop MQSeries, enter the command:

   ```
   endmqm -i QM
   ```

2. Use the installation instructions in Chapter 2 of this manual to install Tivoli Policy Director for MQSeries.

3. To configure the secPKIMap object in LDAP, enter the following command at your command prompt:

   ```
   ldapmodify -h hostname -p 389 -D admin_ID -w admin_password -f
   pdmq_install_path/etc/pdmq.ldif
   ```

   where:

   hostname
   Specifies the host name of the machine running the Lightweight Directory Access Protocol (LDAP) server.

   389 Specifies the LDAP port number. The default is 389.

   admin_ID
   Specifies the LDAP administrator ID that you used when LDAP was installed.

   admin_password
   Specifies your LDAP administrator password.

4. To configure Tivoli Policy Director for MQSeries, enter the command:

   ```
   pdmqcfg -all -admin admin_ID -password admin_password -qm QM
   ```

where:

admin_ID
Specifies your Tivoli Policy Director administrator ID.

admin_password
Specifies your Tivoli Policy Director administrator password.

Note: This command performs several configuration steps. For more information, see Chapter 3, "Configuration" on page 15.

Creating a Tivoli Policy Director User

The following sections use files from the pdmqsamples.zip file, which is located in the /doc directory of the Tivoli Policy Director for MQSeries product CD. After you have installed and configured the MQSeries environment and ensured that it is working, perform the following steps to create a Tivoli Policy Director user:

1. Open the pduser.txt file in a text editor. This file contains the commands that create a single Tivoli Policy Director user and make the user account valid and ready.
2. Modify the company name and country or region code values in the file, as shown:

   ```
   o=company,c=country_or_region_code
   ```

   where:

   company
   Represents your company name as configured in your LDAP server.

   country_or_region_code
   Represents your two-character country or region code.
3. Save and close the pduser.txt file.
4. Use the Tivoli Policy Director **pdadmin** command to create and validate the user:

   ```
   pdadmin -a admin_ID -p password pduser.txt
   ```

   where:

   admin_ID
   Specifies your Tivoli Policy Director administrator ID.

   password
   Specifies your Tivoli Policy Director administrator password.

Creating a Tivoli Policy Director Group

Perform the following steps to create a Tivoli Policy Director group:

1. Open the file pdgroup.txt in a text editor. This file contains the command that creates a single Tivoli Policy Director group.
2. Modify the company name and country or region code values in the file, as shown:

   ```
   o=company,c=country_or_region_code
   ```

   where:

   company
   Represents your company name as configured in your LDAP server.

country_or_region_code
Represents your two-character country or region code.

3. Save and close the pdgroup.txt file.
4. To create a group, use the **pdadmin** command:

```
pdadmin -a admin_ID -p password pdgroup.txt
```

where:

admin_ID
Specifies your Tivoli Policy Director administrator ID.

password
Specifies your Tivoli Policy Director administrator password.

Adding a User to a Group

Perform the following steps to add the Tivoli Policy Director user that you created to the group:

1. Open the pdadduser.txt file, which contains the command that adds a user to a Tivoli Policy Director group.
2. To add your user to your group, use the **pdadmin** command:

```
pdadmin -a admin_ID -p password pdadduser.txt
```

Creating a Tivoli Policy Director Protected Object Policy

Perform the following steps to create a Tivoli Policy Director protected object policy (POP):

1. Open the pdpop.txt file, which contains the command that creates the POP.

   Note: The POP policy sets the quality of protection (QOP) and the audit level. See Chapter 5, "Administering Tivoli Policy Director for MQSeries" on page 33 for more information.
2. To create the POP, use the **pdadmin** command:

```
pdadmin -a admin_ID -p password pdpop.txt
```

Creating the Tivoli Policy Director Access Control List

Perform the following steps to create the Tivoli Policy Director access control list (ACL):

1. Open the pdacl.txt file, which contains the command that sets the group and the permissions.

   Note: For more information about the ACL, see Chapter 5, "Administering Tivoli Policy Director for MQSeries" on page 33.
2. To create the ACL, use the following Tivoli Policy Director **pdadmin** command:

```
pdadmin -a admin_ID -p password pdacl.txt
```

Completing the Administration Process

To complete the administration steps for Tivoli Policy Director for MQSeries, you attach the POP and the ACL. Then you set the extended attributes for the signature algorithm, encryption strength, and recipients to your queue manager and queue. Perform the following steps:

1. Open the QMQmodify.txt file, which contains the commands that attach your ACL to your queue manager, attach the POP to your queues, and assign recipients to the queues. It sets the Q-sig-algorithm attribute to Q–INTEGRITY and the Q-enc-strength and Q-recipients attributes to Q–PRIVACY.

   Note: For more information about extended attributes, see the section on administration in Chapter 5, "Administering Tivoli Policy Director for MQSeries" on page 33.

2. To complete the administration process, use the **pdadmin** command:

   ```
   pdadmin -a admin_ID -p password QMQmodify.txt
   ```

Using a Self-Signed Certificate as a Public Key Infrastructure Identity

To use Tivoli Policy Director for MQSeries, you must create a public key infrastructure (PKI) identity. For this example, use a self–signed certificate, which you create using the Global Security Toolkit (GSKIT) gsk5ikm utility. See the section on managing Public Key Infrastructure identities in Chapter 6, "Managing Identities for Tivoli Policy Director for MQSeries" on page 43 for more information. Perform the following steps:

1. Copy the pdmqkey files (pdmqkey.kdb and pdmqkey.sth) into the *pdmq_install_path*/etc directory.
2. Use the stashed password *test* to open and review the key file pdmqkey.kdb using gsk5ikm. This file contains the self-signed certificate, which uses the label PDMQ Self Signed Cert with the distinguished name cn=pdmqss,o=tivoli,c=us.

Using a Pre-Made Mapping File

Next, you must map your PKI to an operating system user. Tivoli Policy Director for MQSeries performs this mapping through a file named map.conf. See the section "Mapping Operating System Users to Public Key Infrastructure Identities" on page 60 about map.conf. Perform the following steps:

1. If you are using a Solaris or AIX platform, copy the map.conf.unix file as map.conf file to the /opt/pdmq/etc directory, as shown.

   ```
   cp map.conf.unix /opt/pdmq/etc/map.conf
   ```

2. If you are using a Windows NT or Windows 2000 platform, copy the map.conf.intel file as map.conf file to the *pdmq_install_path*/etc directory, as shown:

   ```
   copy map.conf.intel pdmq_install_path/etc/map.conf
   ```

   Note: If your installation path is different from the default, edit the map.conf file to match your installation path.

Performing LDAP Mapping

After you have the operating system identity mapped to the PKI identity, you must map the PKI identity to a Tivoli Policy Director user. This mapping requires you to manipulate Lightweight Directory Access Protocol (LDAP). For more information about user mapping and the Directory Management Tool (DMT), see Chapter 6, "Managing Identities for Tivoli Policy Director for MQSeries" on page 43. In addition to using DMT, you can pass ldif files to the **ldapmodify** command using the instructions provided here. To complete the Tivoli Policy Director for MQSeries user mapping, perform the following steps:

1. Open the uuidfile.ldif file and review the format used to create the Tivoli Policy Director for MQSeries user mapping. You must use this format to complete the

secUUID and the **secCertDN** fields in LDAP in order to specify the Tivoli Policy Director user for which the mapping is needed.

2. Use the following **ldapsearch** command to find the **secUUID** field:

```
ldapsearch -h hostname -D admin_ID -w admin_password -b
 "cn=Users,secauthority=default" secDN=Policy_Director_DN_secUUID
```

where:

Policy_Director_DN_secUUID
Specifies "cn=pdmquser1,o=*company*,c=*country__or_region_code*"

For example:

```
Policy_Director_DN_secUUID is cn=pdmquser1,o=tivoli,c=us
C:\ldapsearch -h hostx -D cn=root -w rootpw -b "cn=Users,secauthority=default"
secDN="cn=pdmquser1,o=tivoli,c=us" secUUID
secUUID=30663b46-ac5f-11d5-a96d-09290192aa77,cn=Users,secAuthority=Default
secUUID=30663b46-ac5f-11d5-a96d-09290192aa77
```

3. Copy the secUUID result for the **ldapsearch** command into the uuidfile.ldif file.
4. Modify the secCertN portion of the uuidfile.ldif file so that it matches the distinguished name of your self-signed certificate.

Note: The final uuidfile file appears as follows:

```
dn: secUUID=30663b46-ac5f-11d5-a96d-09290192aa77,cn=Users,secAuthority=Default
objectclass: SecMap
objectclass: SecPKIMap
secCertDN: CN=pdmqss,o=tivoli,c=US
```

5. To complete the process of mapping the PKI identity to the Tivoli Policy Director user, enter the **ldapmodify** command, using the uuidfile.ldif file:

```
ldapmodify -h hostname –D admin_ID -w admin_password -f uuidfile.ldif
```

6. To verify the mapping, use the **ldapsearch** command:

```
ldapsearch -h hostname -D admin_ID -w admin_password -b
"cn=Users,secauthority=default" secDN=Policy_Director_DN_secUUID
```

The following is a typical example of the results of running the **ldapsearch** command:

```
secUUID=30663b46-ac5f-11d5-a96d-09290192aa77,cn=Users,secAuthority=Default
secdn=cn=pdmquser1,o=tivoli,c=us
secuuid=d79a10ce-bcd9-11d5-8b0c-0004acdd4668
objectclass=SecMap
objectclass=SecPKIMap
objectclass=top
seccertdn=CN=pdmqss,o=tivoli,c=US
```

Updating the Tivoli Policy Director for MQSeries Daemon

To update the mapping information in the Tivoli Policy Director for MQSeries daemon so that you can use the new map.conf file, enter the following:

```
pdmqd -update
```

Using the MQSeries Sample Application to Get and Put Messages from a Queue

After the MQSeries environment is working and Tivoli Policy Director for MQSeries is ready, you can test it using examples with queues of all levels of protection. Enter the following commands:

```
amqsput Q.NONE QM
```

```
amqsget Q.NONE QM
amqsput Q.INTEGRITY.QM
amqsget Q.INTEGRITY.QM
amqsput Q.PRIVACY QM
amqsget Q.PRIVACY.QM
```

To browse the messages after you have placed them on the queue, use one of the following commands:

```
amqsbcg Q.NONE QM
amqsbcg Q.INTEGRITY QM
amqsbcg Q.PRIVACY QM
```

Appendix D. Hardware Accelerator Support

Tivoli Policy Director for MQSeries, Version 3.8.1, uses cryptography to secure the messages. Because cryptographic operations are computing-intensive, they consume most of the CPU time and bring down the whole system performance. This is especially true on a system that conducts a high volume of transactions. This appendix describes the use of a hardware cryptographic acceleration card, which can improve system performance.

Using a Hardware Cryptographic Acceleration Card

The hardware cryptographic acceleration card is designed to offload the cryptographic computing tasks from the main CPU in the system. In addition, it sometimes provides key repository and management functionalities. Although Tivoli Policy Director for MQSeries uses both secret key and public key cryptographic algorithms, the public key cryptographic algorithm uses more CPU time than the secret key algorithm does. This appendix focuses on accelerating the public key algorithm using the following types of cards:

- nCipher nForce 300 SCSI card (Windows 2000 platform only)
- Rainbow CS200, CS600

You can achieve a significant performance gain by adding a hardware acceleration card on a server that processes a large volume of cryptographic operations. However, the performance gain may not be noticeable on a fast system with a small workload. First, you should analyze the system workload, and then put the hardware acceleration card on the system that might be the performance bottleneck. For encryption and decryption purposes, you do not need to have a pair of cards in the systems, because the hardware acceleration card uses the standard algorithms and interoperates with pure software cryptographic systems.

Configuration

By default, the hardware acceleration is disabled. To use the hardware acceleration card with Tivoli Policy Director for MQSeries, you must install the hardware card and configure your system to use the card. This can be done in separate steps. The following are installation options:

- Install and configure card -> Install Tivoli Policy Director for MQSeries -> Configure Tivoli Policy Director for MQSeries
- Install Tivoli Policy Director for MQSeries -> Install and configure card -> Configure Tivoli Policy Director for MQSeries
- Install and configure Tivoli Policy Director for MQSeries -> Stop Tivoli Policy Director for MQSeries -> Install and configure card -> Restart Tivoli Policy Director for MQSeries

For instructions on hardware installation and configuration, refer to the documents shipped with the card.

Enabling Hardware Acceleration

After Tivoli Policy Director for MQSeries and the hardware acceleration card have been installed and configured, an operating system administrator must change the pdmqazn.conf file, which is in the *pdmq_install_path*/etc subdirectory, to enable the use of the card. Add the following entry in the [pdmq] stanza as follows:

```
[pdmq]
acceleratorid=value
```

The value could be one of the the following strings, which are not case-sensitive:

- NCIPHER_NF
- RAINBOW_CS

All other values are regarded as errors. Tivoli Policy Director for MQSeries reports an error during configuration initialization, and the Tivoli Policy Director for MQSeries server (daemon), **pdmqd,** stops.

Only one entry of `acceleratorid` is supported. If multiple entries are in the configuration file, only the first one is used. After the change has been made, you must restart **pdmqd** in order to pick up the new changes.

Disabling Hardware Acceleration

To disable the hardware support, you must stop the daemon, remove or comment out the acceleratorid line in the pdmqazn.conf file, and then restart **pdmqd** in order to pick up the change.

The comment-out line looks like this:

```
[pdmq]
#acceleratorid=value
```

Appendix E. Notices

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

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

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation 2Z4A/101 11400 Burnet Road Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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

COPYRIGHT LICENSE:

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

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

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

AIX

DB2

IBM

IBMLink

IBM logo

MQseries

OS/2

OS/390

RACF

RS/6000

SecureWay

Tivoli

Tivoli logo

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

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

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

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

Index

Q

S

T

U

V

W

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.