

HP OpenView Messaging Server Using Radia

Radia Messaging Server Guide

Software Version: 2.0

for the UNIX and Windows operating systems



Manufacturing Part Number: T3424-90058

August 2004

© Copyright 2004 Hewlett-Packard Development Company, L.P.

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

© Copyright 1998-2004 Hewlett-Packard Development Company, L.P.

No part of this document may be copied, reproduced, or translated into another language without the prior written consent of Hewlett-Packard Company. The information contained in this material is subject to change without notice.

Trademark Notices

Linux is a registered trademark of Linus Torvalds.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Acknowledgements

PREBOOT EXECUTION ENVIRONMENT (PXE) SERVER
Copyright © 1996-1999 Intel Corporation.

TFTP SERVER
Copyright © 1983, 1993
The Regents of the University of California.

OpenLDAP

Copyright 1999-2001 The OpenLDAP Foundation, Redwood City, California, USA.
Portions Copyright © 1992-1996 Regents of the University of Michigan.

OpenSSL License

Copyright © 1998-2001 The OpenSSLProject.

Original SSLeay License

Copyright © 1995-1998 Eric Young (eay@cryptsoft.com)

DHTML Calendar

Copyright Mihai Bazon, 2002, 2003

Technical Support

Please select Support & Services from the following web site:

[<http://www.hp.com/managementsoftware/services>](http://www.hp.com/managementsoftware/services)

There you will find contact information and details about the products, services, and support that HP OpenView offers.

The support site includes:

- Downloadable documentation
- Troubleshooting information
- Patches and updates
- Problem reporting
- Training information
- Support program information

About this Guide

The Radia Messaging Server is a new Radia Infrastructure component that provides a generic routing and inter-server delivery service, especially for report-bound data.

The Radia Messaging Server currently supports the following Radia products:

- Radia Inventory Manager (when configured for Radia Messaging Server support)
- Radia Management Portal
- Radia Patch Manager
- Radia Usage Manager

For Inventory, Operations, and Patch data sent to the Radia Configuration Server, the Radia Messaging Server performs the routing of the data from the Radia Configuration Server to the appropriate external location for each message type.

For Usage Reports, the Radia Messaging Server will route data directly from the clients to reporting aggregation points, and then forward the data to the Radia Knowledge Base.

Who this Guide is for

This guide is for Systems Administrators who are using Radia products that include the Radia Messaging Server, or who want to migrate to the use of the Radia Messaging Server.

What this Guide is about

The *Radia Messaging Server Guide* contains installation, configuration, migration, and tuning information for the Radia Messaging Server.

Conventions

You should be aware of the following conventions used in this book.

Table P.1 ~ Styles

Element	Style	Example
References	<i>Italic</i>	See the <i>Publishing Applications and Content</i> chapter in this book.
Dialog boxes and windows	Bold	The Radia System Explorer Security Information dialog box opens.
Code	Andale Mono	radia_am.exe
↵	Arial Unicode MS	When displaying lines of code that extend beyond the defined margins of the manuscript, this symbol indicates that the code continues uninterrupted and indented on the next line. ADDRESS EDMLINK 'qmsg -priority 20 -to CORE.RIM↵ ZSVCSTAT SESSION PREFACE ZCONFIG ';
Selections	Bold	Click Next to continue.

Table P.2 ~ Usage

Element	Style	Example
Drives (system, mapped, CD)	Italicized placeholder	<i>SystemDrive</i> : \Program Files\Novadigm might refer to C:\Program Files\Novadigm on your computer. <i>CDDrive</i> : \client\radia_am.exe might refer to D:\client\radia_am.exe on your computer.
Files (in the Radia Database)	All uppercase	PRIMARY
Domains (in the Radia Database)	All uppercase	PRIMARY.SOFTWARE May also be referred to as the SOFTWARE domain in the PRIMARY file.
Classes (in the Radia Database)	All uppercase	PRIMARY.SOFTWARE.ZSERVICE May also be referred to as the ZSERVICE class in the SOFTWARE domain in the PRIMARY file.

The table below describes terms that may be used interchangeably throughout this book.

Table P.3 ~ Terminology*

* Depends on the context. May not always be able to substitute.

Term	May also be called
Application	software, service
Client	Radia Application Manager and/or Radia Inventory Manager
Computer	workstation, server
NOVADIGM domain	PRDMAINT domain Note: As of the 4.0 release of the database, the NOVADIGM domain is being renamed the PRDMAINT domain. Therefore, if you are using an earlier version, you will see the NOVADIGM domain in the database.
Radia Configuration Server	Manager, Active Component Server
Radia Database	Radia Configuration Server Database

Contents

Preface	5
About this Guide	5
Who this Guide is for	5
What this Guide is about	5
Conventions	6
 1 Introduction	 13
Defining the Radia Messaging Server	14
Features and Capabilities	16
Benefits over Previous Implementations	16
 2 Installing the Radia Messaging Server on the Radia Configuration Server	 17
Installing the Radia Messaging Server	18
System Requirements	18
Creating a Radia Messaging Server Environment on your Radia Configuration Server	18
Task 1: Create the Data/Default Collection Point	19
Task 2: Verify or Update the Radia Configuration Server Method ZTASKEND	19
Task 3: Verify or Add the Radia Configuration Server Executable QMSG	20
Task 4: Install the Radia Messaging Server	21
Task 5: Verify or Modify the Radia Messaging Server for Radia Patch Manager	36
Task 6: Make Product-Specific Changes	38
Starting and Stopping the Radia Messaging Server	39
Windows Procedures	39
UNIX Procedures	39
Summary	41
 3 Configuring and Tuning the Radia Messaging Server	 43
Understanding the Radia Configuration Server Modules that Support the Radia Messaging Server	44

About the new ZTASKEND method	45
ZTASKEND calls to QMSG	46
Data Collection for Larger Inventory Objects: Fileaudt, WBEM and CLISTATS Objects	48
QMSG Call Syntax.....	48
QMSG Call Parameters:	49
How Priority Establishes Radia Messaging Server Processing Order.....	50
Modifying the Processing Priority.....	51
Configuring the Radia Messaging Server	54
Editing the RMS.CFG File	54
About the Sections in the RMS.CFG File	56
Configuring the Register Default Section.....	57
Configuring the Poll Interval and Post Quantity	58
Configuring for Retry Attempts.....	58
Configuring the Number of Workers	59
Disabling Processing of the /data/default Messages	59
Modifying the Priority in which Messages are Processed.....	60
Configuring the Register Router Section	60
Configuring the Radia Messaging Server to Discard Messages (Set USE as /dev/null)	61
Configuring Radia Messaging Server to Route RMP Messages.....	62
Configuring for Failover	64
Configuring the Log Level, Log Size and Number	65
Changing the Logging Level.....	65
Changing the Size and Number of Log Files	66
Summary	67
 4 Migrating Inventory Processing to use QMSG and the Radia Messaging Server	69
Why Migrate?	70
Migration Benefits	70
Migrating from RADISH to QMSG and the Radia Messaging Server: A Simple Analogy	71
Technical Description	72
Differences in RIM Information Flow using RADISH versus QMSG with the Radia Messaging Server	73
Overview and Alerts!	75
Table of RCS and RMS Components.....	75
Installation Steps on the Radia Configuration Server	77
Upgrading your RIM Server	77
Recommended RIM Server Settings	78
Considering the Impact on Your Production Environment	78
Summary	79

5 Troubleshooting	81
Troubleshooting the Radia Messaging Server	82
Problem:	82
Solution:	82
Problem:	82
Solution:	82
Summary	84
 A Alternative Radia Messaging Server Configurations	 85
Optional Radia Messaging Server Configurations	85
Example 1: Configuring the Radia Messaging Server for Store and Forward	86
Installing and Configuring a "Receiving" Radia Messaging Server	87
Configuring a Radia Messaging Server to Forward Messages to Another Radia Messaging Server	88
Example 2: Configuring Radia Messaging Server to Route to Multiple Queues	89
Example 3: Configuring Radia Messaging Server for Processing Multiple Message Queues	91
 Lists	 93
Figures	93
Tables	95
Procedures	96
 Index	 97



Introduction

At the end of this chapter, you will:

- Be familiar with the Radia Messaging Server.
- Understand how the different Radia products use the Radia Messaging Server.
- Understand the Radia Messaging Server process for queuing and transferring messages.

Defining the Radia Messaging Server

The Radia Messaging Server (RMS) is a generic messaging service that can be used with many Radia Infrastructure modules. Its job is to continually monitor a predefined data queue and dynamically route data objects to one or more external destinations. The Radia Management Server provides retry, rerouting, and failover capabilities to ensure all data is transferred efficiently and reliably.

On a Radia Configuration Server, the Radia Messaging Server operates hand-in-hand with the new executable, QMSG, to handle the transfer of reporting data obtained from clients to the appropriate external Radia Integration Server. From the external Radia Integration Server, the data is mapped to the appropriate ODBC reporting database.

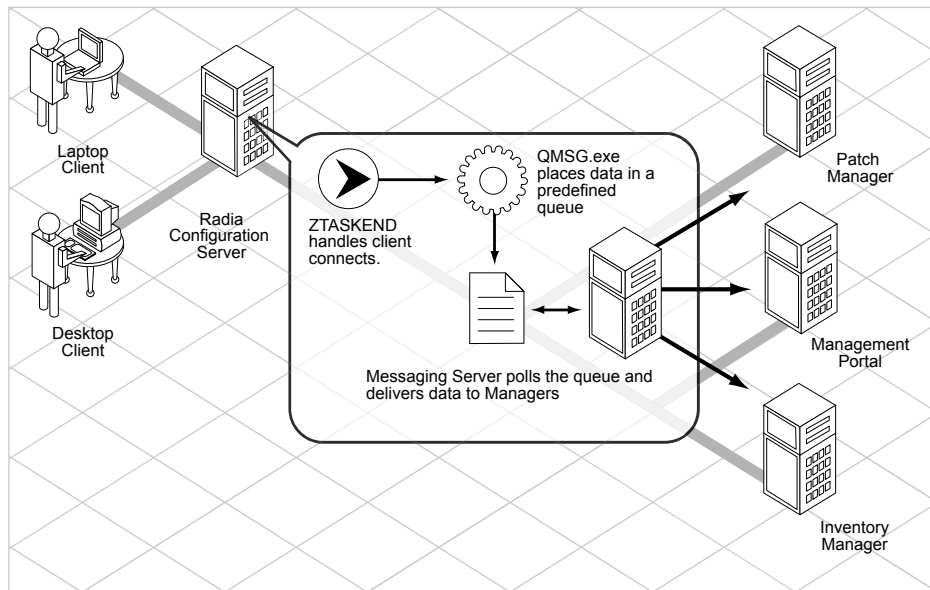


Figure 1.1 ~ Radia Messaging Server process for routing data from RCS to external locations.

The operation works like this:

- The ZTASKEND method calls the QMSG executable to handle all incoming reporting data from clients, such as operations, inventory, and patch data.

Note

Earlier Radia Messaging Server implementations delivered a ZTASKEND method with calls to SENDMSG. As of June of 2004, SENDMSG was renamed QMSG to convey its processing role more accurately and thus ZTASKEND now calls QMSG instead of SENDMSG.

- QMSG places the data objects in a predefined data queue on the Radia Configuration Server.
- The Radia Messaging Server polls the queue and automatically picks up and transfers data objects to the appropriate external locations. The external locations are defined in the rms.cfg configuration file.
- For failover support, more than one Radia Integration Server can be defined as the destination point for each type of data object. If a connection with the first service fails, the reporting data can be rerouted to the failover service.

The Radia Messaging Server runs on all Windows and UNIX platforms supported by the Radia Configuration Server. It communicates with the Radia Integration Server for delivering Inventory, Management Portal, or Patch data using HTTP/XML.

The Radia Messaging Server is configured through the rms.cfg file, which resides in the MessagingServer folder on the Radia Configuration Server. Additional tuning options address load balancing when processing high-volumes of data as well as large-sized objects.

Features and Capabilities

The Radia Messaging Server provides a generic, efficient, and flexible messaging system that can be used by all Radia Infrastructure modules. For example, it can:

- Route a single message to multiple destinations.
- Automatically retry a delivery.
- Re-route messages to a new host after several unsuccessful delivery attempts.

Benefits over Previous Implementations

- Radia Configuration Server performance is enhanced with the use of the Radia Messaging Server, because:
 - The QMSG executable has no startup overhead.
 - The turnaround time of QMSG is faster than RADISH (the executable that QMSG replaces).
 - The job of QMSG executable is greatly reduced:
 - Job of mapping data to SQL is offloaded to the Radia Inventory Manager (RIM) Server.
 - Data is no longer posted directly from the Radia Configuration Server to Radia Inventory Manager server.
 - Bottlenecks on the Radia Configuration Server caused by the processing of reporting data are eliminated.
- Reliability of processing Radia Inventory Manager data is increased, due to:
 - Built-in retry capability.
 - Messages will remain in the queue until they are successfully delivered.
 - Messages remaining in the queue after several delivery attempts can be re-routed to a new host.
 - Retry, holding, and re-routing features eliminate potential loss of data caused by HTTP failures. Summary
- The Radia Messaging Server routes message objects collected from clients to the appropriate RIS server.
- Messages processed include data for operations, inventory, patch, or usage reports.
- Configuration settings in the rms.cfg file allow you to specify the target server and processing priorities by data type. Additional tuning options address load balancing when processing high-volumes of data as well as large-sized objects.

Installing the Radia Messaging Server on the Radia Configuration Server

At the end of this chapter, you will:

- Understand the Radia Messaging Server system requirements.
- Be able to modify the Radia Configuration Server components, as necessary.
- Be able to install the Radia Messaging Server.
- Be able to start and stop the Radia Messaging Server.

Installing the Radia Messaging Server

The following tasks are needed to provide initial support for a Radia Messaging Server environment on a Radia Configuration Server. The mechanics of the installations and modifications are provided in this chapter.

System Requirements

- The Radia Messaging Server can be installed on any Windows or UNIX platform running the Radia Configuration Server.

Creating a Radia Messaging Server Environment on your Radia Configuration Server

Perform the following tasks to install the Radia Messaging Server on your Radia Configuration Server. The same tasks can be followed whether you are installing it for the first time, or upgrading from an earlier version.

1. Create the Data/Default collection point for messages on the Radia Configuration Server.
2. Verify or update the version of the ZTASKEND REXX method on the Radia Configuration Server.
3. Verify or add the QMSG.EXE executable on the Radia Configuration Server.
4. Install the Radia Messaging Server.
5. If using Radia Patch Manager, verify or modify the product-specific modifications for patch.
If you need to change the DSN user password entry after running the original install, this task also explains how to create an encrypted password entry for the Radia Messaging Server configuration file.
6. Make product-specific modifications to Radia Inventory Manager modules.

Note

The Radia Messaging Server includes store and forward capabilities. If you are taking advantage additional Radia Messaging Servers to be included in your environment.

Task 1: Create the Data/Default Collection Point

Create a new **data/default** directory under the Radia Configuration Server directory. This directory is where the QMSG executable places messages to be picked up and routed by the Radia Messaging Service.

FOR EXAMPLE

- If the Radia Configuration Server is installed on a Windows machine at:

`C:\Novadigm\ConfigurationServer`

Create:

`C:\Novadigm\ConfigurationServer\data\default`

- If the Radia Configuration Server is installed on a UNIX machine at:

`/opt/Novadigm/ConfigurationServer`

Create:

`/opt/Novadigm/ConfigurationServer/data/default`

Task 2: Verify or Update the Radia Configuration Server Method ZTASKEND

The Radia Messaging Server works hand-in-hand with the version of the ZTASKEND REXX method that includes calls to the QMSG executable. This version of ZTASKEND was released in June of 2004, and is distributed with Radia Configuration Server media created after that date.

If you have a Radia Configuration Server with a ZTASKEND dated prior to June of 2004, use the procedure *To replace the ZTASKEND method*, which follows, to replace the ZTASKEND method on your Radia Configuration Server with the required version.

Note

For additional details regarding ZTASKEND and the calls to QMSG that collect data for the Radia Messaging Server, see *About the new ZTASKEND method* on page 45.

To replace the ZTASKEND method

1. Hewlett-Packard recommends making a backup of any modules being replaced. These are listed in Table 4.2.
2. Hewlett-Packard also recommends stopping the service for the Radia Configuration Server to make these changes. However, if your Radia Configuration Server machine is operating close to 100% CPU usage, it will be **necessary** to stop the service in order to replace ZTASKEND.
3. Copy the new ZTASKEND from the \media\rexx folder of the Radia Configuration Server source media to the **rexx** directory of your Radia Configuration Server.

- For Windows, the default is: C:\Novadigm\ConfigurationServer\rexx.
- For UNIX, the default is: /opt/Novadigm/ConfigurationServer/rexx.

Note

ZTASKEND is frequently customized. If your ZTASKEND that calls RADISH is customized, you'll need to port the custom code to the new ZTASKEND that calls QMSG. You can place a copy of the ZTASKSEND that calls QMSG (prior to customizations) in the **rexx/Novadigm** directory of the Radia Configuration Server. After porting your customizations, place the customized ZTASKEND that calls QMSG in the **rexx** directory of the Radia Configuration Server.

4. The service for the Radia Configuration Server can be restarted after you add QMSG.EXE in the next step.

Task 3: Verify or Add the Radia Configuration Server Executable QMSG

The Radia Configuration Server executable QMSG is required for this release of the Radia Messaging Server. QMSG became available in June of 2004, and is distributed with Radia Configuration Server media released after that date.

Check the following location to see if QMSG.EXE exists on your Radia Configuration Server:

Windows: <RCS directory>\bin
UNIX: <RCS directory>/exe

- If QMSG.EXE exists on your Radia Configuration Server, continue with the next task: *Task 4: Install the Radia Messaging Server.*
- If your Radia Configuration Server does not include QMSG.EXE, use the following procedure to add it for Radia Messaging Server support.

To add QMSG.EXE to the Radia Configuration Server

1. Copy **QMSG.exe** from the appropriate <platform> directory of the Radia Configuration Server media to your Radia Configuration Server's **bin** directory (if Windows) or **exe** directory (if UNIX).
 - For Windows, the default is: C:\Novadigm\ConfigurationServer\bin.
 - For UNIX, the default is: /opt/Novadigm/ConfigurationServer/exe.
2. Restart the service for the Radia Configuration Server.

Task 4: Install the Radia Messaging Server

This task installs the Radia Messaging Server on the same machine as your Radia Configuration Server. During the install, you are prompted to specify the following items related to message pickup and delivery locations for your environment:

- The directory to scan for messages.
This is the directory specified in Task 1 on page 19. Normally, the scan directory is the \data\default directory of where your Radia Configuration Server is installed.
- The store & forward port number for receiving messages from another Radia Messaging Server.
- The IP Address and Port for routing Radia Inventory Manager data.
- The IP Address and Port for routing Radia Management Portal data.
If you are not routing data to a Radia Management Portal, leave these fields blank and the Radia Messaging Server will automatically discard any Radia Management Portal data it finds in the scan directory.
- A DSN, user and password for connecting to a SQL database.
If you are routing data to an ODBC database, such as Radia Patch Manager data, enter the connection information. Otherwise, leave these blank.

Note

If you are updating an existing Radia Messaging Server, the install procedure will update the program files, but will **not** overwrite or make any updates to an existing configuration file, rms.cfg. This allows any previous customization settings in rms.cfg to remain intact.

If you want to create a new rms.cfg file based on installation dialog entries, rename your rms.cfg file to rms.bak before running the install procedure. Following installation, you can edit the new rms.cfg file to port any customizations from the rms.bak file to the rms.cfg file.

To run the install for the Radia Messaging Service

1. Launch the installation program for the Radia Messaging Server, available from the following platform-specific location on the Radia Infrastructure CD:

```
\extended_infrastructure\messaging_server\<platform>
```

- For Windows, click on **setup.exe** to launch the installation program.
- For a UNIX platform, enter the following command:

```
./install
```

and press **Enter**.

The **Welcome** window opens.

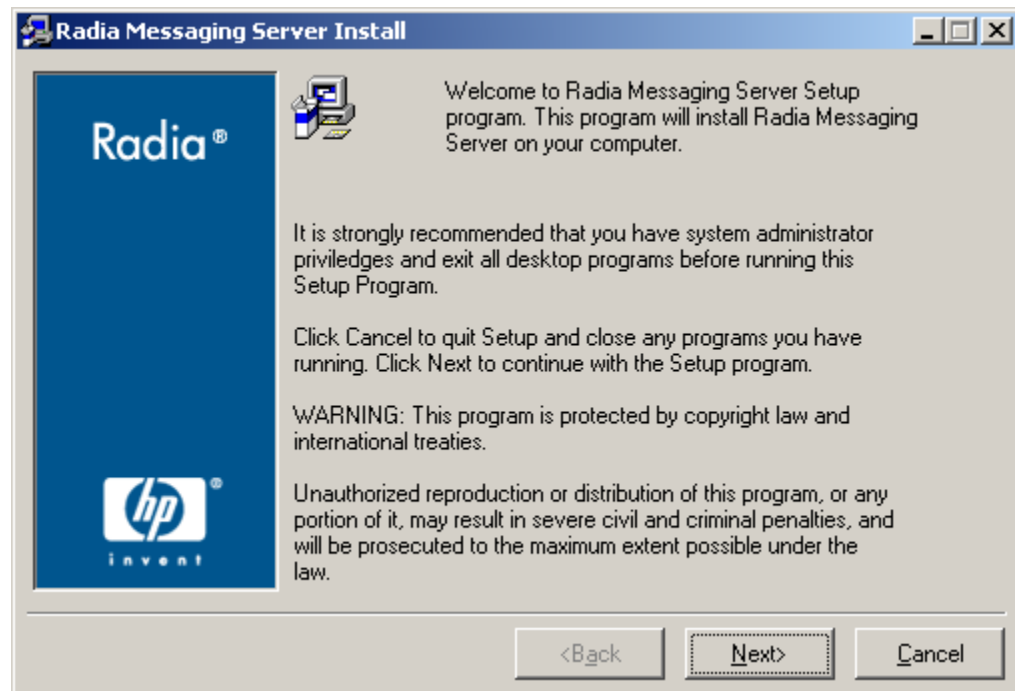


Figure 2.1 ~ Radia Messaging Server welcome window.

2. Click Next.

The **End-User License Agreement** window opens for you to read the licensing terms for this product. You must accept the terms before the Radia Proxy Server can be installed.

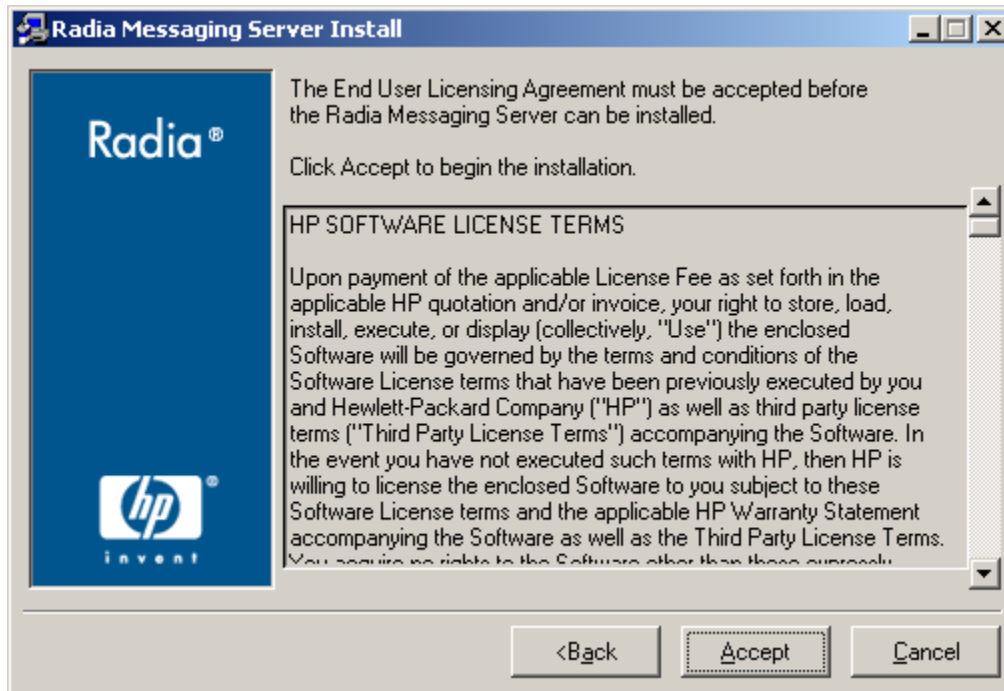


Figure 2.2 ~ End User Licensing Agreement.

3. Click **Accept** to agree to the terms of the software license and continue with the installation. The **Installation Location** window opens.

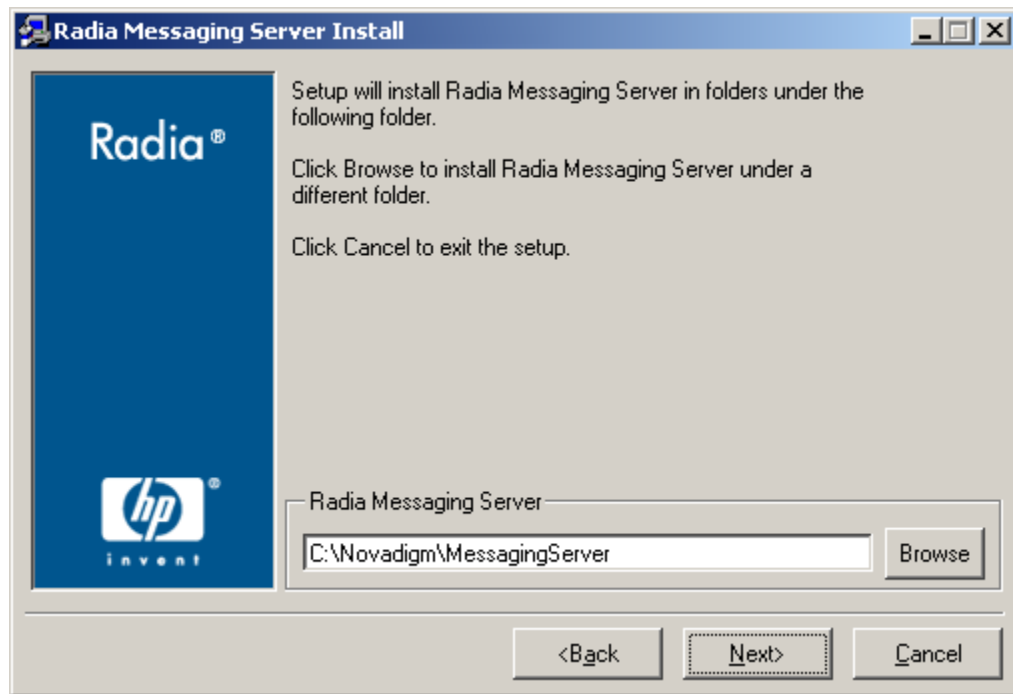


Figure 2.3 ~ Radia Messaging Server location window.

4. Select the location to install the Radia Messaging Server. Accept the default location or type a location that is on the same machine as the Radia Configuration Server. Use the **Browse** button to manually select the location.

5. Click **Next**.

The **Settings for the Radia Inventory Manager** window opens.



Figure 2.4 ~ Type the IP Address of the Radia Inventory Manager Server.

6. Type the IP address or DNS host name of the Radia Inventory Manager Server, and click **Next**.

The Radia Messaging Service will route all inventory data to this server.

Note

Following installation, you can modify the `rms.cfg` file to add failover processing for an alternate Radia Inventory Manager server. See *Configuring for Failover* on page 64 for more information.

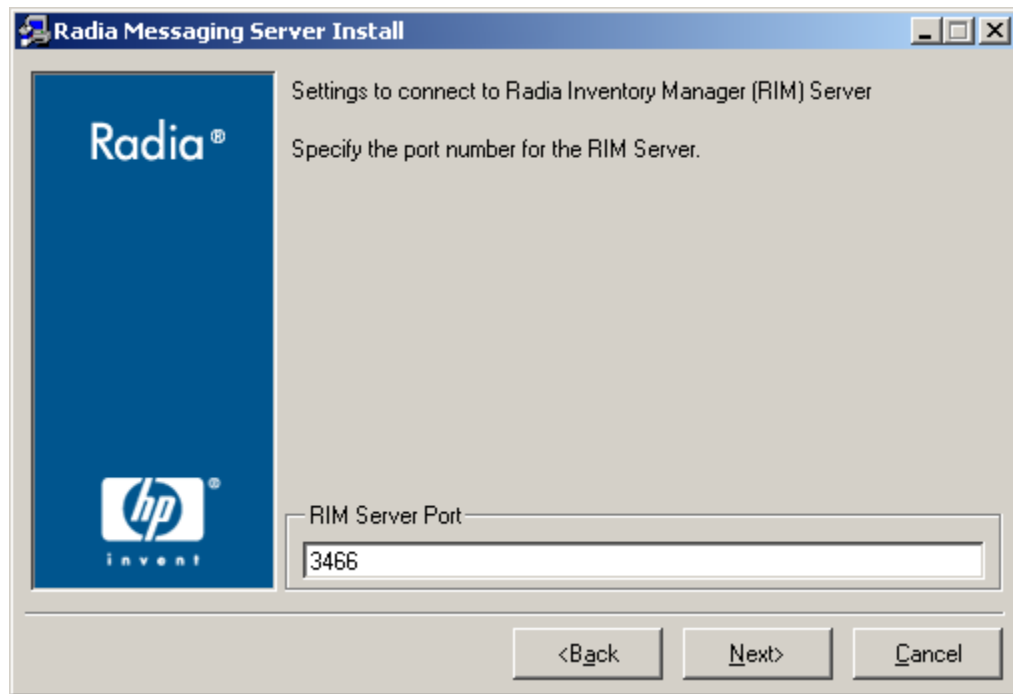


Figure 2.5 ~ Type the port number of the Radia Inventory Manager Server.

7. Type the port number of the Radia Inventory Server. Normally, this is 3466.
8. Click **Next**.

The **Settings for the Radia Management Portal** window opens.



Figure 2.6 ~ Type the IP Address of the Radia Management Portal Server.

9. Type the IP address or DNS host name of the Radia Management Portal server to route all RMP data found in the scan directory location to that server.

OR

To automatically discard any Radia Management Portal data found in the scan directory location, leave the RMP IP Address field **blank**.

10. Click **Next**.

The **Radia Management Portal Port** window opens.

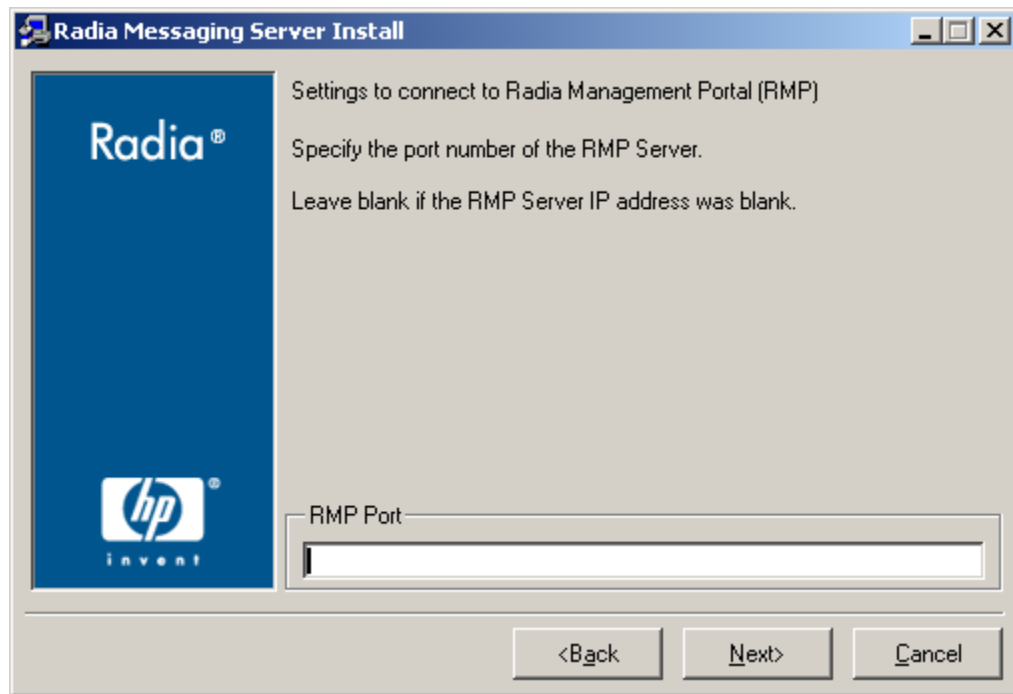


Figure 2.7 ~ Port number for the Radia Management Portal.

- 11.** If you entered an RMP IP Address on the previous window, type the port number of the Radia Management Portal. Normally, this is 3466.
OR
Leave the **RMP Port** blank if you also left the RMP IP Address field blank.
- 12.** Click **Next**.
The **Store & Forward Port Setting** window opens.

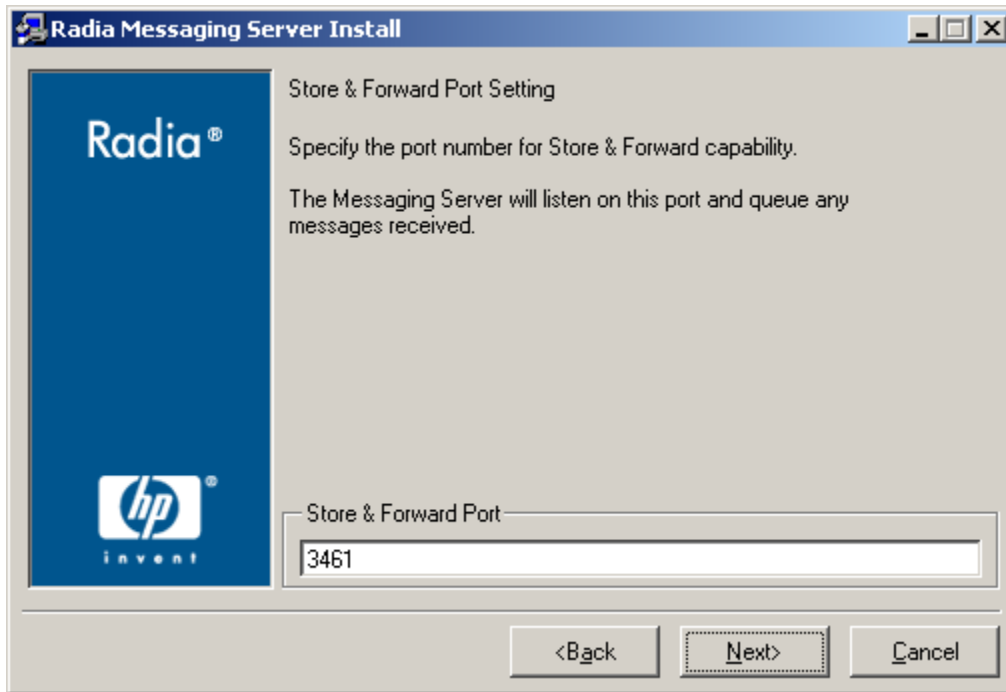


Figure 2.8 ~ Store & Forward Port number to receive forwarded messages.

The Radia Messaging Server includes the ability to receive and process messages that have been forwarded from other Radia Messaging Servers in your enterprise. The port used to receive these messages is called the store & forward port.

For more information on using store & forward, see *Example 1: Configuring the Radia Messaging Server for Store and Forward* on page 86 of *Appendix A: Alternative Radia Messaging Server Configurations*.

13. Accept the default store & forward port, **3461**, or type another port number to use to receive any messages from other Radia Messaging Servers.
14. Click **Next**.

The **Data Source Name** window for the ODBC Delivery Settings opens. If the Radia Messaging Server is to route messages directly to an ODBC Data Source, such as the Patch SQL Database, specify the ODBC settings to connect to a SQL database on the next three windows.

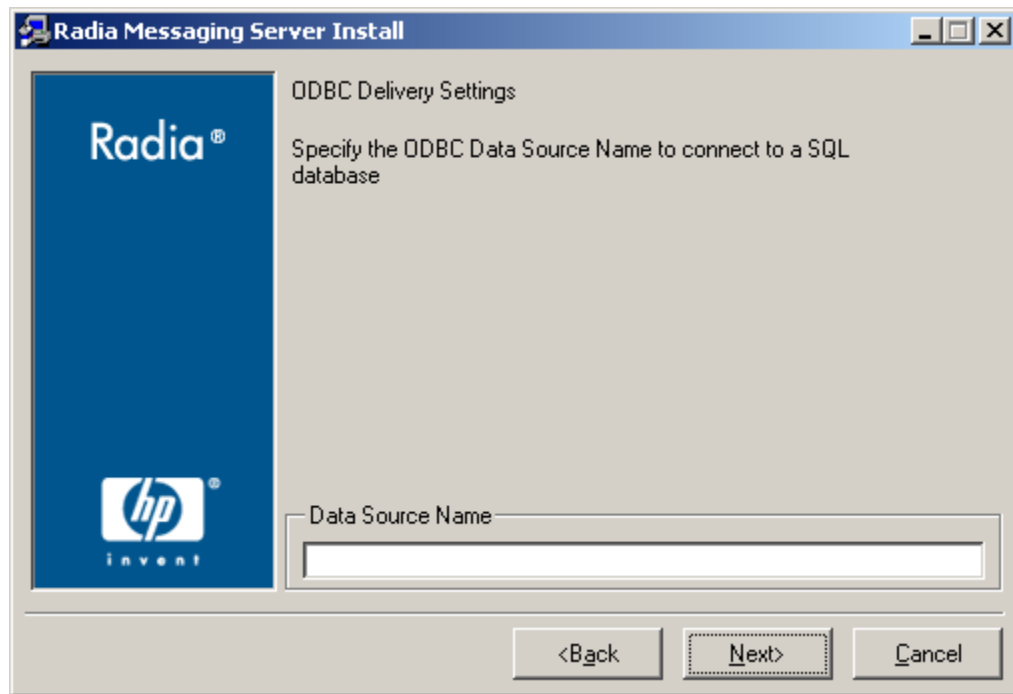


Figure 2.9 ~ If connecting to a SQL Database, such as Patch, specify an ODBC Data Set Name.

- 15.** Type the Data Source Name if the Radia Messaging Server is to connect to an ODBC SQL database, such as for the delivery of Patch messages.
Leave blank if an ODBC connection is not required.
- 16.** Click **Next**.
The **DSN User Name** window for the ODBC Delivery Settings opens.

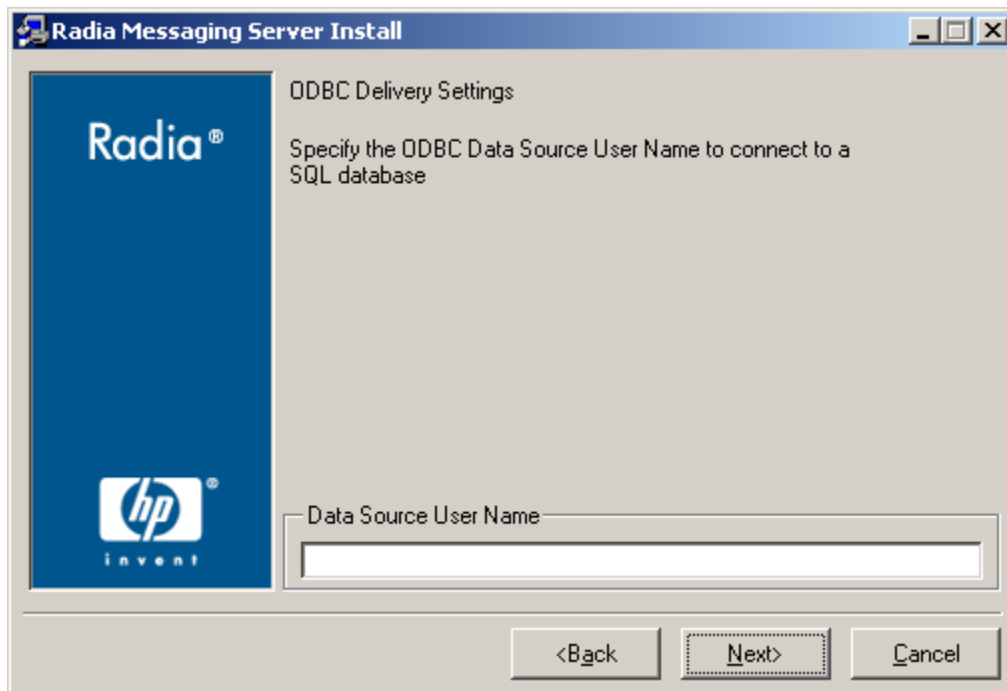


Figure 2.10 ~ If connecting to a SQL Database, such as Patch, specify the Data Set User Name.

17. Type the DSN User Name to use to connect to an ODBC SQL database.
Leave blank if an ODBC connection is not required.
18. Click **Next**.
The **DSN Password** window for the ODBC Delivery Settings opens.



Figure 2.11 ~ If connecting to a SQL Database, such as Patch, specify the Password for the DSN User.

19. Type the password required for the DSN user entered on the previous dialog.
Leave blank if an ODBC connection is not required.

Note

The password will be encrypted.

20. Click **Next**.
The **Message Directory to Scan** window opens.

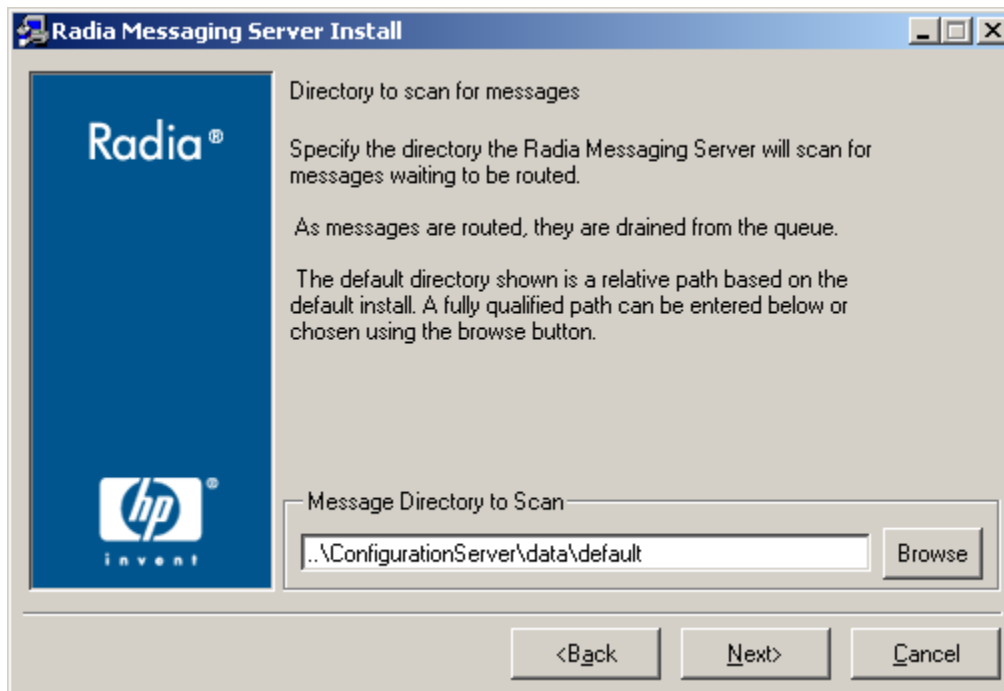


Figure 2.12 ~ Specify the directory to scan for messages on the Radia Configuration Server.

- 21.** Accept the default or use the browse button to select the directory where the Radia Messaging Server should scan for messages. Specify the directory created in *Task 1: Create the Data/Default Collection Point* on page 19. Normally, this is the **\data\default** folder of where the Radia Configuration Server is installed.
- 22.** Click **Next**.

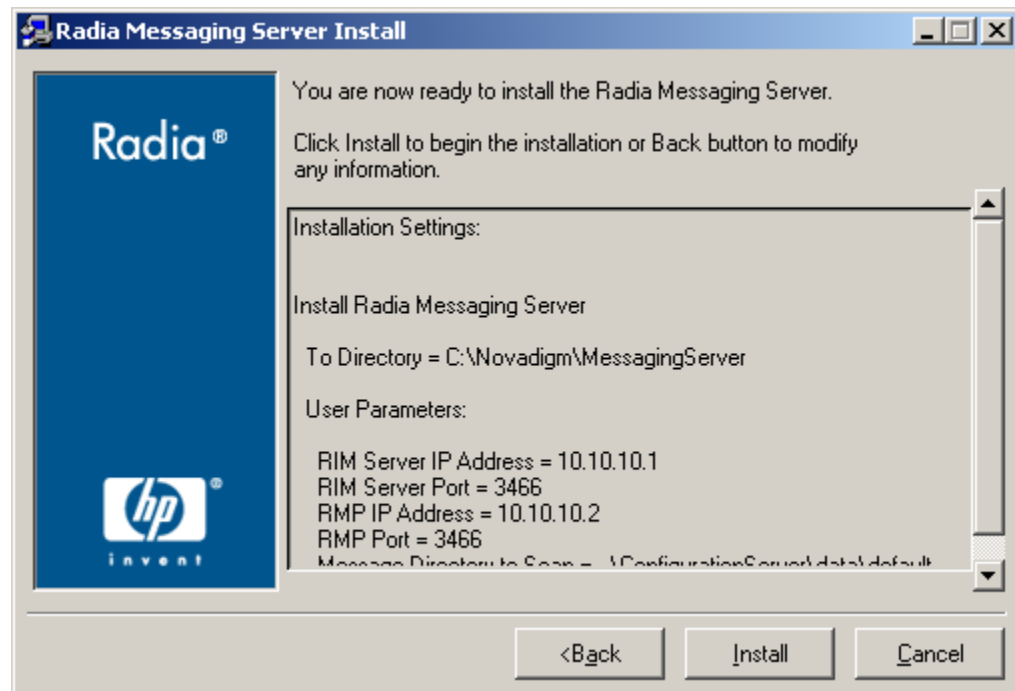


Figure 2.13 ~ Verify the summary.

23. Verify the summary screen and click **Install**.

Read and answer any warning dialogs that appear. Which dialog boxes appear will depend on your configuration.

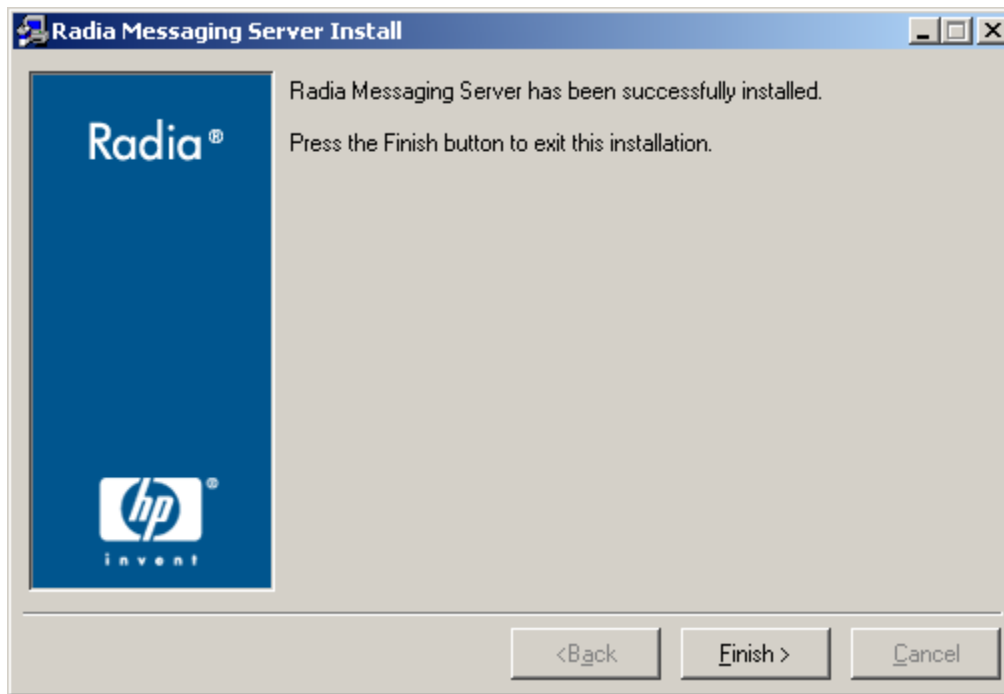


Figure 2.14 ~ The installation is complete.

24. Click **Finish**.

The Radia Messaging Service has been installed and configured for routing Radia Inventory Manager, Radia Management Portal, and, optionally, connecting to and routing messages for Radia Patch Manager to an SQL Database (if entered).

Once started, the Radia Messaging Server scans the newly created **data/default** directory on the Radia Configuration Server for message files and delivers the messages to the specified destinations.

The log file for the Radia Messaging Server (rms.log) resides in the **Logs** folder of the **MessagingServer** directory. For example: C:\Novadigm\MessagingServer\Logs for Windows, or /opt/Novadigm/MessagingServer/Logs for UNIX.

- To modify a routing for Radia Patch Manager data, see the next task.
- To update RIM.TKD to work with the Radia Messaging Server, see *Task 6: Make Product-Specific Changes* on page 38.

Note

See *Chapter 3: Configuring and Tuning the Radia Messaging Server* starting on page 43 for additional configuration options. Tuning options include changing the polling frequency or retry value, and specifying failover servers for inventory data.

Task 5: Verify or Modify the Radia Messaging Server for Radia Patch Manager

If you are using Radia Patch Manager, use the procedure below to verify the lines needed by the Radia Messaging Server to support Patch as a router and to register its ODBC database.

In addition, you should also check the *Radia Patch Manager Guide* for the latest information.

To modify/verify the rms.cfg file entries for Patch Manager

1. Stop the Radia Messaging Server service. See *Starting and Stopping the Radia Messaging Server* on page 39 for more information.
2. Use a text editor to open **rms.cfg** and review the following patch-related entries exist in the rms.cfg file. If necessary, make the following changes in each step that follows. The file is located in the directory where you installed the Radia Messaging Server. The default location is:
For windows: <System Drive>\Novadigm\MessagingServer
For UNIX: opt/Novadigm/MessagingServer
3. Find the section starting with "msg::register router," and, if necessary, add the lines in the rectangle in Figure 2.15 below.

```

msg::register router {
  TYPE      ROUTER

  ROUTE     {
    TO      CORE.RIM
    USE     rim
  }
  ROUTE     {
    TO      CORE.RMP
    USE     /dev/null
  }
  ROUTE     {
    TO      INVENTORY
    USE     rim
  }
  ROUTE     {
    TO      INVENTORY.WBEM
    USE     rim
  }
  ROUTE     {
    TO      PATCH
    USE     patch
  }
}

```

Verify/add these lines.

Figure 2.15 ~ For Patch support, add lines to the Register Router section.

4. Go to the end of the rms.cfg file, and look for the lines shown in Figure 2.16. If necessary, add these lines to the end of the rms.cfg file.

```

msg::register patch {
  TYPE      ODBC

  DSN       "put your dsn here"
  USER      "user"
  PASS      "password"
}

```

Figure 2.16 ~ For Patch Support, add these lines to the end of the rms.cfg file.

5. Specify the values for the DSN, USER, and PASS values for access to the Patch SQL database, as follows:
 - DSN** – Specify the Data Source Name (DSN) for the Patch SQL database. Enclose the DSN in quotation marks.
 - USER** – Specify the SQL user for the DSN. Enclose the user name in quotation marks.
 - PASS** – Specify the password for the SQL user for the DSN. Enclose the password entry in quotation marks. To obtain an encrypted password value, use the procedure *To encrypt the password entry for the Patch SQL database DSN in rms.cfg* on page 38.

6. Save your changes and exit the editor.
7. Start the Messaging Service, rms.tkd. For details, see *Starting and Stopping the Radia Messaging Server* on page 39.

To encrypt the password entry for the Patch SQL database DSN in rms.cfg

The PASS value in the msg::register patch section of the rms.cfg file specifies the password for the SQL user for the DSN for the Patch SQL database. It is encrypted during the setup.exe installation program for security purposes. If you need to modify the password, you can use the **nvdkit** utility to create an encrypted password, and specify this encrypted value within the msg::register patch section of the rms.cfg file. Enclose the encrypted value in quotation marks.

1. Open a command prompt and go to the directory where the Messaging Server is installed.
2. Enter the following command:
nvdkit
3. At the % prompt, type the following command:
password encrypt <password_value>
The utility will return an encrypted password value.
4. Cut and paste this encrypted password value into the rms.cfg file as the PASS value. Enclose the value in quotation marks. The PASS value is located in the msg::register patch section as shown in Figure 2.16 on page 37.

Task 6: Make Product-Specific Changes

After completing the installation steps on the Radia Configuration Server, you may need to upgrade the RIM Server with a new rim.tkd.

For details, see the topics, *Upgrading your RIM Server* on page 77 and *Recommended RIM Server Settings* on page 78.

Starting and Stopping the Radia Messaging Server

Use the procedures that apply to your type of operating system:

- Windows procedures start on page 39.
- UNIX procedures start of page 39.

Windows Procedures

The Radia Messaging Server is automatically installed as a Windows service. The service name is **Radia Messaging Server (rms)**.

- Use the **Services** window of your operating system to start or stop the Radia Messaging Server.
- Alternatively, to start or stop the installed service from a command prompt, open a DOS window and type the following commands from the \MessagingServer directory:

```
nvdkit rms.tkd start  
nvdkit rms.tkd stop
```

Once the Windows service for the installed Radia Messaging Server is stopped, you can run it from a command prompt:

- Open a DOS window and type the following command from the \MessagingServer directory on your Radia Configuration Server machine:

```
nvdkit rms.tkd
```

- To stop a Messaging Service running in a DOS window, make the window active and press CTRL+C.

UNIX Procedures

- To start the Radia Messaging Service, go to the /MessagingServer directory on your Radia Configuration Server machine and type the following command to run it in the background:

```
./nvdkit rms.tkd &
```

To run the Radia Messaging Service in the foreground, omit the '&' in the previous command.

- To stop the Radia Messaging Service, go to the /MessagingServer directory on your Radia Configuration Server machine. First obtain its Process ID (PID) and then kill the process.

Note

The following are general guidelines and the commands are examples that may vary slightly depending on the UNIX type you are using.

To obtain the PID for the Radia Messaging Service, type the following command to list all the UNIX processes for nvdkit:

```
ps -f | grep nvdkit
```

Run the following command to kill the PID listed for the Radia Messaging Server.

```
kill -9 <PID>
```


Summary

- To create a Radia Messaging Server environment on your Radia Configuration Server, you need to have the ZTASKEND REXX method with calls to QMSG, the QMSG executable, and then install the Radia Messaging Server.
- The Radia Messaging Server is installed as a Windows service or a UNIX process.
- If you are using Patch Manager, you may need to edit rms.cfg file to enable Patch Manager support.
- You may also need to update files on your Radia Inventory Manager server to support the Radia Messaging Server.

Configuring and Tuning the Radia Messaging Server

At the end of this chapter, you will:

- Be able to configure the Radia Configuration Server method and executables that support the Radia Messaging Server.
- Be able to configure the parameters in `rms.cfg`.
- Be able to configure the Radia Messaging Server for failover.
- Be able to tune the configuration to adjust for heavy loads.

Note

- The first part of this chapter discusses the Radia Configuration Server modules that support the Radia Messaging Server.
- The topics for configuring and tuning the Radia Messaging Server begin on page 54.

Understanding the Radia Configuration Server Modules that Support the Radia Messaging Server

This topic explains how the ZTASKEND method and QMSG executable work hand-in-hand with the Radia Messaging Server to collect, queue, and then deliver data to the appropriate external location.

You should be familiar with the topics presented here before customizing ZTASKEND.

Note

Earlier Radia Messaging Server implementations delivered a ZTASKEND method with calls to SENDMSG. As of June of 2004, SENDMSG was renamed QMSG to convey its processing role more accurately and thus ZTASKEND now calls QMSG instead of SENDMSG.

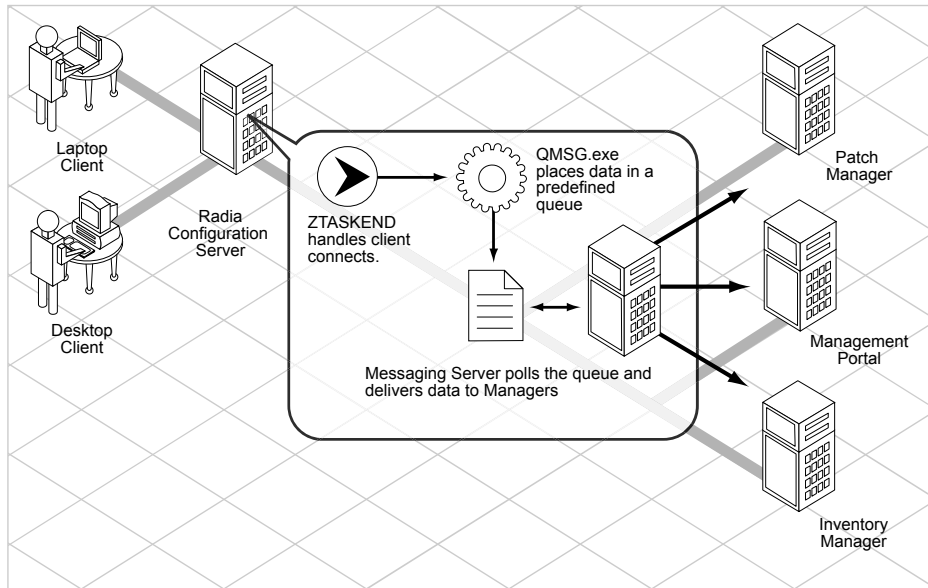


Figure 3.1 ~ ZTASKEND invokes QMSG, which places data in a location for Radia Messaging Server processing.

About the new ZTASKEND method

The ZTASKEND method on the Radia Configuration Server is called at the end of each client connection, while storage and objects associated with the connection are still available. ZTASKEND calls QMSG when client data needs to be collected for another service, such as RIM for Radia Inventory Report data, or RMP for Radia Management Portal data. QMSG collects the data and places the messages in the /data/default location for pickup and processing by the Radia Messaging Server.

An important job of ZTASKEND is to ensure unique client data is collected at the appropriate client connect phase.

This topic explains:

- How ZTASKEND determines when to call QMSG for the various client connect phases and which objects are collected.
- The syntax of calls to QMSG.
- How the QMSG -priority parameter establishes Radia Messaging Server processing order.
- How the QMSG -to parameter establishes one or more destinations for message processing.

Note

This topic reflects the ZTASKEND method delivered with the Radia 4 release and Radia Messaging Server 2.0.

ZTASKEND calls to QMSG

An important role of ZTASKEND is to **minimize the collection of duplicate information**. With that goal in mind, the ZTASKEND method invokes QMSG using the following logic:

1. Whenever one of the following critical objects is presented:
 - Application Event
 - MSI Event
 - Synopsis object
2. If a critical object is not present, then ZTASKEND calls QMSG when a client connects during the following phases:
 - Catalog resolution (for Radia Inventory Manager [RIM] and Radia Management Portal [RMP] collection)
 - Client reporting phases (for Inventory collection, only)
3. If a critical object is not found, the ZTASKEND method does *not* invoke QMSG for the following client phases:
 - Client Operations Profiles (COP) resolution
 - Single service resolution
 - Client maintenance resolution
4. Finally, ZTASKEND does not invoke QMSG to obtain error message objects.
5. The objects that are collected vary by client phase. Refer to the lines of code near the bottom of ZTASKEND to identify which objects are collected for a given client phase.

```

if PHASE = 'CATALOG_RESO' then objects = 'SESSION PREFACE ZSTATUS ZCONFIG ZMASTER SMINFO'
if PHASE = 'CLIENT_REPORTING' then objects = 'SESSION PREFACE ZSTATUS SAPSTATS ZRSTATE SMINFO'
if PHASE = 'BOOTSTRAP' then objects = 'SESSION PREFACE ZSTATUS SMINFO'
if PHASE = 'SERVICE_RESO' then objects = 'SESSION PREFACE ZSTATUS SMINFO'
if PHASE = 'CLIENT_SELFMAINT' then objects = 'SESSION PREFACE ZSTATUS SMINFO'

```

Figure 3.2 ~ Sample ZTASKEND code identifying objects collected by QMSG calls, by client phase.

The following table summarizes the Client Connect phases and the objects collected during the ZTASKEND calls to QMSG for operational, CORE.RIM, and CORE.RMP data.

Table 3.1 ~ ZTASKEND calls to QMSG for CORE.RIM and CORE.RMP Data, by Client Connect Phase

Client Connect Phase	QMSG call if no critical object	QMSG call if critical object
COP Resolution: for Client Operations Profile.	<i>No</i>	Collects these objects for CORE.RIM and CORE.RMP destinations: APPEVENT MSIEVENT SYNOPSIS SESSION PREFACE ZSTATUS SMINFO
Client Maintenance Phase:	<i>No</i>	Collects these objects for CORE.RIM and CORE.RMP destinations: APPEVENT MSIEVENT SYNOPSIS SESSION PREFACE ZSTATUS SMINFO
Catalog Resolution: Client connects to the Radia Configuration Server to obtain service resolution list.	Always. Collects these objects for CORE.RIM and CORE.RMP destinations: SESSION PREFACE ZCONFIG ZMASTER ZSTATUS SMINFO	See previous column, but also collects APPEVENT MSIEVENT SYNOPSIS.
Single Service Resolution: For each service to be resolved, client makes another connection to the Radia Configuration Server.	<i>No</i>	Collects the following objects for CORE.RIM and CORE.RMP destinations: APPEVENT MSIEVENT SYNOPSIS SESSION PREFACE ZSTATUS SMINFO
Client Reporting Phase: At the end of service resolution. client data is reported back to the Radia Configuration Server.	Always. Collects these objects for CORE.RIM destination: SESSION PREFACE ZSTATUS SAPSTATS ZRSTATE SMINFO	See previous column, but also collects APPEVENT MSIEVENT SYNOPSIS.

Data Collection for Larger Inventory Objects: Fileaudt, WBEM and CLISTATS Objects

In the latter half of ZTASKEND is the code to collect the objects for FILEAUDT, Inventory.WBEM, and CLISTATS objects. The code and calls to QMSG to obtain these objects are shown in the following figure. The calls to QMSG are in bold.

```

/* Post the fileaudt object to RIS if present */
CALL EDMGET 'FILEPOST',1 ;
if FILEPOST \= 'FILEPOST' then do
    if debug > 0 then say '***<>*** Queuing INVENTORY objects for delivery -->
FILEPOST'
    ADDRESS EDMLINK 'qmsg -priority 20 -to INVENTORY FILEPOST';
end;
/* Post the wbemaudt AND/or CLISTATS objects to RIS if present */
CALL EDMGET 'WBEMAUDT',1 ;
CALL EDMGET 'CLISTATS',1 ;
MGR_RIM_PARM = ''
IF CLISTATS \= 'CLISTATS' THEN MGR_RIM_PARM = 'CLISTATS '
IF WBEMAUDT \= 'WBEMAUDT' THEN MGR_RIM_PARM = MGR_RIM_PARM || 'WBEMAUDT '
if MGR_RIM_PARM \= '' then do
    if debug > 0 then say '***<>*** Queuing INVENTORY.WBEM objects for delivery --
> ' MGR_RIM_PARM
    ADDRESS EDMLINK 'qmsg -priority 20 -to INVENTORY.WBEM ' MGR_RIM_PARM';
end;

```

Figure 3.3 ~ ZTASKEND code with QMSG calls to collect larger Inventory objects: Fileaudt, WBEMAUDT, and CLISTATS objects.

The call syntax includes the `-priority` parameter to establish a lower processing priority for these large objects. See *How Priority Establishes Radia Messaging Server Processing Order* on page 50 for more information.

QMSG Call Syntax

The base syntax for QMSG calls is discussed below. [Brackets] indicate optional entries.

Note

The phase-related calls to QMSG you see at the bottom of ZTASKEND substitute variables for the call parameters. This topic explains the base syntax behind a QMSG call.

```
Format: 'qmsg [-priority nn] -to DEST1[,DEST2] OBJECT1 [OBJECTn]' | OBJVAR
```

```
Example 1:      'qmsg -to CORE.RIM,CORE.RMP
```

```
Example 2:      'qmsg -priority 20 -to INVENTORY FILEPOST'
```

```
Example 3:      'qmsg -priority 20 -to INVENTORY.WBEM ' MGR_RIM_PARM
```

Figure 3.4 ~ Sample syntax and examples of ZTASKEND calls to QMSG.

The syntax allows each message placed in the `./data/default` directory to include a processing priority and 'to' destination. Messages going to multiple locations will have more than one destination. The QMSG parameters are discussed below.

QMSG Call Parameters:

-priority

is optionally coded to establish Radia Messaging Server processing priority. If omitted, a default priority of 10 is given to the message. Valid values are 00 (highest priority) to 99 (lowest priority). For more information on Radia Messaging Server processing priority, see the topic *How Priority Establishes Radia Messaging Server Processing Order* on page 50.

-to DESTINATION(S)

must be explicitly coded with one or more destinations. Messages going to multiple destinations have comma-separated entries. For example:

```
-to CORE.RIM,CORE.RMP
```

Each message destination requires an equivalent ROUTE defined for it in the `msg::register` router section of the `rms.cfg` file, as illustrated in Figure 3.9 on page 61. The current destination values used by the delivered QMSG include:

```
-to CORE.RIM
```

```
-to CORE.RIM,CORE.RMP (these messages are delivered to both destinations)
```

```
-to INVENTORY
```

```
-to INVENTORY.WBEM
```

Note

If you are adding a QMSG call for messages going `-to PATCH`, you will also need to code an equivalent ROUTE for the PATCH destination in the `rms.cfg` file. For more details, see the topic *Task 5: Verify or Modify the Radia Messaging Server for Radia Patch Manager* on page 36.

OBJECT1 [OBJECTn]

entries immediately following the `-to` destinations name the Radia objects collected. The object list is space separated. A sample object list includes the bold items in the following call:

```
'qmsg -to CORE.RIM SESSION PREFACE ZSTATUS'
```

OBJVAR

Example 3 in Figure 3.4 shows an alternative QMSG call format using a variable instead of an explicit object list.

```
'qmsg -priority 20 -to INVENTORY.WBEM ' MGR_RIM_PARM
```

In this example, the variable `MGR_RIM_PARM` is coded after the closing quotation mark. It is used to collect `WBEMAUDT` and/or `CLISTATS` objects. Notice the `-priority 20` is also coded in this example, to establish a lower priority than the other objects. For details, see the following topic.

How Priority Establishes Radia Messaging Server Processing Order

When `ZTASKEND` calls `QMSG`, the `-priority` parameter in the call assigns a processing priority to the message. Priority values can range from 00 to 99, with 00 reserved for critical processing and 99 being the lowest priority.

Figure 3.5 illustrates how priority affects processing order: the Radia Messaging Server will process all messages assigned to a higher priority (such as 10) *before* processing any messages assigned to a lower priority (such as 20). Within a given priority, messages are processed using first in, first out (FIFO) order.

Note

The message priority stays the same for the life of the message. For example, if a message is forwarded from one Radia Messaging Server to another, the message priority remains the same.

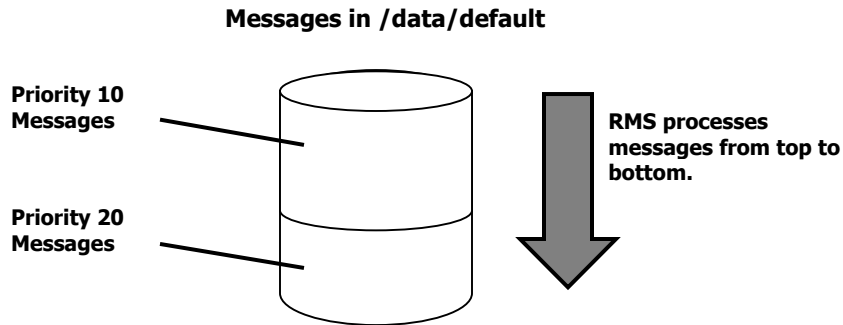


Figure 3.5 ~ Messages are sorted in ascending priority, and then processed top down by Radia Messaging Server.

As of this writing, ZTASKEND is coded as follows:

- QMSG assigns most messages a priority of 10. Priority 10 is the default if **-priority** is not specified.
- QMSG assigns a lower priority of 20 to the larger objects collected for Inventory Reports: these include file audits, whem reporting data, and client statistics (CLISTATS). A sample call to QMSG using priority 20 is given below:

```
ADDRESS EDMLINK 'qmsg -priority 20 -to INVENTORY FILEPOST';
```

If the Radia Messaging Server is not able to process the messages as fast as they are delivered from QMSG, the lower priority messages will accumulate at the bottom of the /data/default location, even though newer messages with higher priorities are still being processed.

Warning

If the Radia Messaging Server cannot drain the queue completely, messages at the bottom may never be processed. To increase the rate at which messages are processed, please refer to the *topic Configuring the Register Default Section* on page 57.

Modifying the Processing Priority

Novadigm has assigned processing priorities to meet the needs of most customers. Table 3.2 below categorizes the objects processed by QMSG and their given priorities.

Table 3.2 ~ ZTASKEND Calls to QMSG by Object Type and Priority

Object Type	To parameter value	Contains Objects	Priority (default is 10)
RIM Reporting	CORE.RIM	RIM Session and Preface Objects, possibly AppEvent	10
RMP Data	CORE.RMP	RMP Session and Preface Objects, possibly AppEvent.	10
RIM Reporting	INVENTORY	FILEPOSTS	20 (Lower)
RIM Reporting	INVENTORY.WBEM	WBEM Reporting Data and Client Statistics (WBEMAUDT and CLISTATS)	20 (Lower)
Patch	PATCH	Patch objects: PREFACE and ZOBJSTAT	10

If you have a different need, however, you can modify the **–priority** parameter in ZTASKEND calls to QMSG to accommodate it.

Warning

Modifying priorities is a substantial change. We recommend testing the change in a non-production environment with sufficient message quantities before adopting it in production.

To modify a QMSG priority assignment

1. Place a backup copy of the current ZTASKEND that calls QMSG (prior to customizations) in the **rexx/Novadigm** directory of the Radia Configuration Server.
2. Use a text editor to open ZTASKEND.
3. Locate the QMSG call for the object type whose priority you want to change. For example, to change the priority for processing WBEM objects, locate the following code near the end of ZTASKEND:

```
say '**<>** Queuing INVENTORY.WBEM objects for delivery';
ADDRESS EDMLINK 'qmsg -priority 20 -to INVENTORY.WBEM ' MGR_RIM_PARM;
end;
```



To increase priority, change 20 to 10.

Figure 3.6 ~ Edit the QMSG call's –priority parameter to change processing priority by object type.

4. Replace the value of the **–priority** parameter with a number from 00 to 99. The lower the number, the higher the processing priority. If omitted, a priority of 10 is assigned.
For example, to give WBEM objects a higher processing priority, replace **–priority 20** with **–priority 10** in the line of code illustrated in Figure 3.6 .
5. Save your changes and close the file.
6. Restart the Radia Configuration Server.

Configuring the Radia Messaging Server

Use these topics to reconfigure or tune the Radia Messaging Server after installation.

Editing the RMS.CFG File

You can adjust the default values by editing the rms.cfg file, located in the \etc directory of where the Radia Messaging Server was installed on your Radia Configuration Server machine. Table 3.3 defines the rms.cfg parameters, along with their default values and their definitions.

To edit the rms.cfg file and parameters

1. Stop the Messaging Server before editing the rms.cfg file. For details, see *Starting and Stopping the Radia Messaging Server* on page 39.
2. Edit the rms.cfg file using any text editor. By default, rms.cfg is located at:
<SystemDrive>\Novadigm\MessagingServer\etc for Windows, or
/opt/Novadigm/MessagingServer/etc for UNIX.
3. Modify the sections using the information given in the following topics.

Important!

All path entries in the rms.cfg file must be specified using forward slashes. This applies to both Windows and UNIX environments.

4. Save your changes and restart the Radia Messaging Server.

```

# DO NOT REMOVE THE NEXT LINE
package require nvd.msg
log::init {
    -loglevel 3
}
# ATTEMPTS * DELAY = Maximum time in seconds an item will remain in the queue
# 200 * 3600 = ~8 days
msg::register default {
    TYPE        QUEUE
    DIR          ../ConfigurationServer/data/default
    USE          router
    POLL         10
    COUNT        100
    DELAY        3600
    ATTEMPTS     200
    WORKERS      1
}
msg::register httpd {
    TYPE        HTTPD
    PORT        3461
    USE         default
    URL         /proc/rim/default
    URL         /proc/xml/obj
}
msg::register router {
    TYPE        ROUTER

    ROUTE       {
        TO       CORE.RIM
        USE      rim
    }
    ROUTE       {
        TO       CORE.RMP
        USE      /dev/null
    }
    ROUTE       {
        TO       INVENTORY
        USE      rim
    }
    ROUTE       {
        TO       INVENTORY.WBEM
        USE      rim
    }
    ROUTE       {
        TO       PATCH
        USE      patch
    }
}
msg::register rim {
    TYPE        HTTP

    ADDRESS     {
        PRI     10
        URL     http://111.111.11.1:3466/proc/rim/default
    }
}
msg::register rmp {
    TYPE        HTTP

    ADDRESS     {
        PRI     10
        URL     http://localhost:3466/proc/xml/obj
    }
}
msg::register patch {
    TYPE        ODBC

    DSN         ""
    USER        ""
    PASS        "{DES}:0"
}

```

Figure 3.7 ~ Typical rms.cfg, configured to deliver RIM messages and discard RMP messages.

Note

The Radia Messaging Server must be stopped and restarted for changes to the rms.cfg file to take effect.

About the Sections in the RMS.CFG File

The rms.cfg file has the following main sections after the header.

- **log::init**
Sets the Logging Level for entries written to the log files. The default is 3. Normally, this is not changed. For details on changing the logging level, see *Configuring the Log Level, Log Size and Number* on page 65. The log files are located in the Logs directory of the root MessagingServer installation directory.
- **msg::register default**
Defines how the Radia Messaging Server handles the messages placed by QMSG in the /data/default location (or /data/default queue). The parameters are defined in Table 3.3, and summarized below:
 - DIR defines the full path of the /data/default location. This is set at installation time.
 - USE defines where the routing information for each TO label is located.
 - POLL and COUNT establish the polling interval and post quantity for the Radia Messaging Server, which determines how often and how many messages are posted at a time. To adjust this, see the topic *Configuring the Poll Interval and Post Quantity*.
 - Retry Attempts (DELAY and ATTEMPTS), after maximum retry attempts, a message is automatically discarded
 - WORKERS parameter (optional). If not specified, a default of one WORKER is used. You can add the following line to increase the number of WORKERS:

WORKERS 2

 Set to 2, or up to 4. You can disable processing by setting WORKERS to -1.
- **msg::register router**
Configures routing assignments for each -To type. This section enables you to route messages to more than one destination. It also allows you to route messages to a set disposal location of /dev/null. At least one route is specified for each -To type:
 - -To inventory
 - -To inventory.wbem
 - -To rim.core
 - -To rmp.core
 - -To patch
- **msg::register <USE type>** (for example: msg::register rim, msg::register rmp, etc.)

- Initial definition of HTTP locations established during installation.
 - Configure for Failover.
 - Configure to Discard Messages.
- (Optional) Configure the maximum log size and number of logs. Note that these options apply for each Worker assigned to process the /data/default queue. See Configuring the Log Size and Number for more information. See for *Configuring the Log Level, Log Size and Number* on page 65 details.

Configuring the Register Default Section

Use the following table to modify the parameters in the msg::register default section of rms.cfg.

Table 3.3 ~ RMS.CFG Parameters used to Define the /Data/Default Queue

Parameter	Default	Definition
TYPE	QUEUE	Registration type. <i>Do not change this value.</i>
DIR	../ConfigurationServer/data/default	Directory where your Radia Configuration Server (through QMSG.exe) will queue XML objects to post. Edit the DIR value to reflect the full path of your data/default directory <i>using forward slashes</i> for both Windows and UNIX platforms.
USE	router	Internal setting telling the program what process to use. <i>Do not change this setting.</i>
POLL	10	Delay in seconds for polling the local store of objects to be posted. Increase this value to support the posting of large objects, such as those for Operational reports.
COUNT	100	How many XML objects will be posted at each POLL interval.
DELAY	3600	Amount of time in seconds to retry a failure.
ATTEMPTS	200	How many times the Radia Messaging Server will try to post a message before discarding it. Note: DELAY * ATTEMPTS gives the maximum time a message will stay in the queue before automatic discard. Using the default values of DELAY and ATTEMPTS, a message is discarded after approximately 8 days of failed posting attempts.

Table 3.3 ~ RMS.CFG Parameters used to Define the /Data/Default Queue

Parameter	Default	Definition
WORKERS	1	<p>Optional entry. Number of asynchronous, lightweight processes to create for this queue.</p> <p>To drain a queue more quickly, we recommend using WORKERS set to 2. A second worker doubles the processing power of a single Radia Messaging Server configuration, with each worker performing a separate POLL and COUNT.</p> <p>Using more than 4 WORKERS is NOT recommended.</p> <p>Note: Set to -1 (minus 1) to temporarily disable a queue from being processed.</p>

Configuring the Poll Interval and Post Quantity

By default, the Radia Messaging Server is configured to poll the /data/default location every 10 seconds, and post up to 100 objects at a time. To change the poll interval, modify the POLL parameter in rms.cfg; to change the maximum number of objects to be posted at a time, modify the COUNT parameter in rms.cfg.

If the objects being posted are very large, we suggest increasing the POLL interval to give sufficient time to complete the posting.

Configuring for Retry Attempts

The Radia Messaging Server is configured to retry any message that fails to post. By default, the Radia Messaging Server will retry posting the message every hour, and make up to 200 attempts. These values are defined by the DELAY and ATTEMPTS entries in the **msg::register default** section of rms.cfg.

Important Note

After the last attempt, the message is automatically discarded from the queue without being posted.

To calculate the maximum time that a message could stay in the /data/default queue, take the DELAY time and multiply it by the ATTEMPTS value. Using the default settings, this is a DELAY of 3600 seconds x 200 ATTEMPTS, or approximately eight days.

You can adjust the DELAY and ATTEMPTS values in rms.cfg to establish a different maximum time that a message could stay in the /data/default queue. Specify the DELAY in seconds.

Configuring the Number of Workers

By default, the Radia Messaging Server starts one asynchronous process to attend the /data/default message directory.

To create multiple processes or to temporarily disable processing of the messages by the Radia Messaging Server, add a WORKERS parameter to the **msg::register default** section with the appropriate value.

- To increase the number of WORKERS, we recommend setting WORKERS to 2, but not more than 4. A second worker doubles the processing power of a single Radia Messaging Server configuration, with each worker performing a separate POLL and COUNT Detailed steps are given in the following procedure.
- To temporarily disable processing of the queue, set WORKERS to -1 (minus 1). Disabling WORKERS is rarely used.

To add a WORKERS value to create multiple processes (WORKERS)

1. Use a text editor to open **rms.cfg**.
2. Locate the section starting with 'msg::register default', and add the line shown in **bold** in Figure 3.8.
3. Save your changes and exit the editor.

```

msg::register default {
  TYPE      QUEUE
  DIR       C:/Novadigm/ConfigurationServer/data/default
  USE       router
  POLL      10
  COUNT     100
  DELAY     3600
  ATTEMPTS  200
  WORKERS  2
}

```

← Add this line to increase WORKERS.

Figure 3.8 ~ Edit the DIR value in the 'msg::register default' section of rms.cfg file.

Disabling Processing of the /data/default Messages

The objects in a disabled queue are not polled or posted. You may want to disable processing during peak client connect periods if resources are in contention, or if you know a target server is down.

You can re-enable the processing at night or during slower periods to allow the Radia Messaging Server to transfer the messages.

To disable processing

To disable a queue from being polled and its contents posted, add the `WORKERS` to the `msg::register default` section of `rms.cfg` with a value of `-1` (minus 1).

To enable a disabled queue

To enable a previously disabled queue, change the `WORKERS` value in the `msg::register default` section of `rms.cfg` from `-1` to a positive number. The number of `WORKERS` indicates the number of independent and lightweight processes to be started for this queue. The default configuration uses 1. To increase processing, we recommend using 2, but not more than 4.

Modifying the Priority in which Messages are Processed

To modify the priority in which messages are processed, see the earlier topics *How Priority Establishes Radia Messaging Server Processing Order* on page 50 and *Modifying the Processing Priority* on page 51.

Configuring the Register Router Section

The Router section defines at least one `ROUTE` for each `-to` destination value placed in the queue by `QMSG`.

The messages in the `/data/default` location are sent there with a `-to` destination that identifies their contents. The `-to` labels include:

- `CORE.RIM`
- `CORE.RMP`
- `INVENTORY`
- `INVENTORY.WBEM`

There may also be others, such as

- `PATCH`

The register router section must define one or more posting locations for each message with a given `-to` value. Figure 3.9 shows a `ROUTE` is defined for each `-to` value passed from `QMSG`.

The routes for `CORE.RIM`, `INVENTORY`, and `INVENTORY.WBEM` all use the same posting location of `'rim'`. As delivered, the route for `CORE.RMP` uses a special location that discards those messages (`/dev/null`). For more details, see *Configuring the Radia Messaging Server to Discard Messages* on page 61.

```
msg::register router {
  TYPE      ROUTER

  ROUTE     {
    TO       CORE.RIM
    USE      rim
  }
  ROUTE     {
    TO       CORE.RMP
    USE      /dev/null
  }
  ROUTE     {
    TO       INVENTORY
    USE      rim
  }
  ROUTE     {
    TO       INVENTORY.WBEM
    USE      rim
  }
  ROUTE     {
    TO       PATCH
    USE      patch
  }
}
```

Figure 3.9 ~ msg::register router section defines a ROUTE for each QMSG – to destination.

Configuring the Radia Messaging Server to Discard Messages (Set USE as /dev/null)

The location of **/dev/null** is built into the messaging server for discarding messages. When **USE** is set to **/dev/null** in any of the rms.cfg sections, the messages being processed will be successfully discarded without an error.

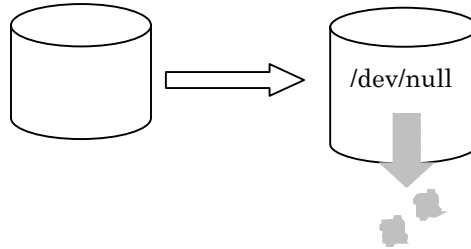


Figure 3.10 ~ Set USE as /dev/null to successfully discard certain messages, or drain a queue.

For example, to discard all messages placed in the Radia Messaging Server queue with a TO label of CORE.RMP, specify the following ROUTE in the msg::register router section of rms.cfg:

```
ROUTE      {
            TO      CORE.RMP
            USE      /dev/null
        }
```

Note

As of this writing, the delivered version of rms.cfg is configured to discard messages for CORE.RMP, as shown above. It is enabled by specifying USE as **rmp**. For details, review *Task 5: Verify or Modify the Radia Messaging Server* for Radia Patch Manager on page 36.

As another example, to quickly drain the entire messaging service queue, temporarily replace **USE router** in the msg::register default section of rms.cfg (as shown in Figure 3.8 on page 59) with **USE /dev/null**. After draining the queue, reset it back to **USE router**.

Configuring Radia Messaging Server to Route RMP Messages

If you initially installed the Radia Messaging Server to discard Radia Management Portal messages, use the steps below to begin routing Radia Management Portal data to an RMP Server and Port.

To modify rmp.cfg for routing RMP data

1. Use a text editor to edit the rms.cfg file.
2. Locate the section starting with '**msg:register router**', and then find the entry for the route defining CORE.RMP messages. Data that is being discarded will show a route entry of 'USE /dev/null'.

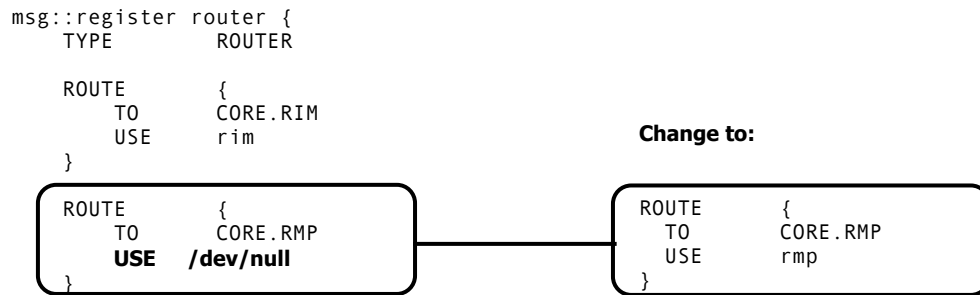


Figure 3.11 ~ Enable RMP message processing by changing USE value to rmp.

On the line below 'TO CORE.RMP', change

USE /dev/null to rmp

as shown in Figure 3.11

3. Next, locate the section starting with '**msg::register rmp**' near the end of the rms.cfg file. Within the section is a URL entry pointing by default to the localhost and port 3466. Change the host and port for the RMP URL to the corresponding host and port of your RMP server. Host names can be specified using an IP address or a DNS name.

```

msg::register rim {
  TYPE      HTTP

  ADDRESS    {
    PRI      10
    URL      http://111.111.111.11:3466/proc/rim/default
  }
}
msg::register rmp {
  TYPE      HTTP
  ADDRESS    {
    PRI      10
    URL      http://localhost:3466/proc/xml/obj
  }
}

```

Specify the host and port of your RMP server in the URL.

↓

localhost:3466

Figure 3.12 ~ Edit the URL value in rmp.cfg to specify the host and port for your RMP server.

4. Save your changes and restart the Radia Messaging Server. For details, see *Starting and Stopping the Radia Messaging Server* on page 39.

To modify rmp.cfg for RMP 1.x

1. If you are running a version of RMP prior to 2.0, first ensure RMP processing is enabled and the URL specifies the appropriate host name and port of the Radia Management Portal. See *To modify rmp.cfg for routing RMP data* on page 62 for more information.
2. For RMP 1.x, also edit the URL entry in the '**msg::register rmp**' section of rmp.cfg to specify a different /proc location; change the URL value for RMP routing from:

URL `http://<hostname:port>/proc/rmp/default`

to

URL `http://<hostname:port>/proc/xml/obj`

```

msg::register rim {
    TYPE      HTTP

    ADDRESS   {
        PRI    10
        URL     http://111.111.111.11:3466/proc/rim/default
    }
}
msg::register rmp {
    TYPE      HTTP

    ADDRESS   {
        PRI    10
        URL     http://111.111.111.21:3466/proc/rmp/default
    }
}

```

For RMP 1.x, change the URL /proc location.



`proc/rmp/default`

Figure 3.13 ~ For RMP 1.x, also edit the URL value in rmp.cfg to specify /proc/rmp/default.

3. After editing the file, save your changes and restart the Radia Messaging Server. For details, see *Starting and Stopping the Radia Messaging Server* on page 39.

Configuring for Failover

You can configure the Radia Messaging Server to have a one or more Radia Inventory Manager Servers defined for failover support, with an assigned PRI value. If the Radia Messaging Server fails to connect with the first Radia Inventory Manager Server (that is, the server with the lowest PRI value), it will try the next server on the list (or, the next higher PRI value).

Note

This PRI value is separate from the `-priority` value assigned by QMSG for processing priority. They are not related.

To set failover in the rms.cfg

Failover support is added by inserting additional ADDRESS entries to the appropriate section of the rms.cfg file. Find the section using Table 3.4:

Table 3.4 ~ Radia Messaging Server Sections for Adding Failover

To provide failover for:	Add ADDRESS entries to section:
Inventory	msg::register rim
Radia Management Portal	msg::register rmp
Patch	msg::register patch

- The URL entries will be tried in order of PRI (priority) starting with the *lowest* PRI value. Figure 3.14 shows sample modifications to the msg::register rim section of rmp.cfg for failover. The code in **bold** was added to define a failover server for Radia Inventory Manager processing.

```

msg::register rim {
  TYPE      HTTP
  ADDRESS    {
    PRI      10
    URL      http://rim1:3466/proc/rim/default
  }
  ADDRESS    {
    PRI      20
    URL      http://rim2:3466/proc/rim/default
  }
}

```

Figure 3.14 ~ Sample rms.cfg file modified for RIM failover. Code in bold added for failover.

Configuring the Log Level, Log Size and Number

The log files for the Radia Messaging Server (rms.log) resides in the **Logs** folder of the **MessagingServer** directory. For example: C:\Novadigm\MessagingServer\Logs.

Changing the Logging Level

The **log::init** section at the beginning of the rms.cfg file establishes the logging level. The default logging level is 3. Valid levels are 0 (no logging) to 10 (maximum logging level). Normally, this is

not changed unless requested by a customer support person for troubleshooting purposes. The following lines show the log level increased to 4:

```
log::init {  
    -loglevel 4  
}
```

Changing the Size and Number of Log Files

The Radia Messaging Server writes entries to a set of log files for each WORKER. By default, it will create and retain up to seven log files per worker, each file having a maximum of 5000 lines. The log files are located in the Radia Messaging Server **\Logs** directory.

To control the size and number of logs created for each worker, add the following entries below the `log::init` section of the `rms.cfg` file (also see Figure 3.15):

```
log.configure -lines <maximum_lines>  
log.configure -size <maximum number of logs>
```

where `<maximum_lines>` is the maximum number of lines for a given log file. After the maximum is reached, another log file is created, until the `<maximum number of logs>` specified in the `log.configure -size` entry is reached. After the `<maximum number of logs>` is reached, the oldest log files are deleted.

Figure 3.15 illustrates an `rms.cfg` file containing entries to limit each log file to 1000 lines, and allow up to 10 log files to be retained.

```
log::init {  
    -loglevel 3  
}  
  
log.configure -lines 1000  
log.configure -limit 10  
  
# ATTEMPTS * DELAY = Maximum time in seconds an item will remain in the queue  
# 200 * 3600 = ~8 days  
  
msg::register default {
```

Figure 3.15 ~ Log.configure lines added to rms.cfg to limit log size and number.

Summary

- The ZTASKEND method calls the QMSG executable to place the appropriate data from client connects in a waiting /data/default directory. The Radia Messaging Server scans this location and posts the messages to the appropriate external locations.
- The parameters in rms.cfg allow you to tune the Radia Messaging Server.
- To adjust for heavy loads, you can add a second WORKER process, adjust the POLL interval, or change the COUNT of the maximum number of objects to be posted at any time.
- To provide failover support, insert multiple ADDRESS entries with varying priority values to the appropriate section of the rms.cfg file. The addresses will be tried in order of PRI (priority) starting with the *lowest* PRI value.
- The Radia Messaging Server is configured to retry any message that fails to post—every hour for up to 200 attempts. To adjust these retry values use the DELAY and ATTEMPTS entries in the **msg::register default** section of rms.cfg.
- The built-in location of /dev/null can be used to discard messages that do not need to be posted.
- If necessary, you can temporarily disable Radia Messaging Server processing by setting the WORKERS value to -1.

Migrating Inventory Processing to use QMSG and the Radia Messaging Server

At the end of this chapter, you will:

- Understand the benefits of migrating an existing Inventory implementation from using RADISH to using QMSG with the Radia Messaging Server.
- Understand the differences and impacts of making the change.
- Know how to make the change.

About Migrating Your Inventory Processing

You now have the option of using the Radia Messaging Server with a new executable, QMSG, to post Radia Inventory Manager information from the Radia Configuration Server to the Radia Inventory Manager Server. This new method replaces the use of the Radia Configuration Server executable, RADISH.

Why Migrate?

Migrating to the use of QMSG with the Radia Messaging Server offers substantial Radia Configuration Server performance benefits as well as posting reliability benefits. Migrating is highly recommended on a go-forward basis to all Radia Inventory Manager customers.

Migration Benefits

The benefits of migrating your inventory processing from using the RADISH executable to using the QMSG executable with the Radia Messaging Server are listed below.

- Radia Configuration Server performance is enhanced. This is because:
 - The QMSG executable has no startup overhead.
 - The turnaround time of QMSG is faster than RADISH.
 - The job of the new QMSG executable is greatly reduced
 - ◆ Data is now mapped to SQL on the Radia Inventory Manager server.
 - ◆ Data is no longer posted directly from Radia Configuration Server to the Radia Inventory Manager server.
 - Bottlenecks on the Radia Configuration Server caused by the processing of reporting data are eliminated.
- Reliability of processing inventory data is increased. This is due to:
 - Radia Messaging Server's built-in retry capability.
 - Messages remain in the data queue until they are successfully delivered.
 - Messages remaining after several delivery attempts can be re-routed to a new host.
 - Retry, holding, and re-routing features eliminate potential loss of data caused by HTTP failures.

Migrating from RADISH to QMSG and the Radia Messaging Server: A Simple Analogy

The ways in which RADISH versus QMSG and Radia Messaging Server handle message processing within the Radia Configuration Server can be compared to the following types of package delivery services.

RADISH – A ONE-PACKAGE AT A TIME COURIER SERVICE

RADISH can be compared to a specialized courier service that picks up and delivers one package at a time. That is, whenever the Radia Configuration Server receives a client *package* for the reporting database, another RADISH courier is called to pick it up. The courier must deliver the package to the RIM server destination before handling another one. If client packages start coming in quickly, there will be a large group of RADISH couriers waiting to delivery their packages (one-by-one) to the same RIM destination. This can result in overcrowding, delays, and possibly lost packages.

QMSG AND RADIA MESSAGING SERVER – A COMBINATION OF COURIER, MAILROOM AND OUTSIDE DELIVERY SERVICE

In contrast, the new system of posting messages using QMSG and Radia Messaging Server can be compared to a system using internal couriers, a mailroom, and an outside package delivery service. This system is designed to collect, sort, and route large volumes of packages with minimal overhead.

For example, the only job of QMSG couriers is to pick up an incoming message and deliver it to the appropriate bin in the mailroom. From there, the agents from the Radia Messaging Server come in on a regular basis to handle the pickup, routing, and delivery of the pre-sorted messages from the appropriate bin in the mailroom, and deliver them to the RIM Server destinations. Radia Messaging Server agents will retry delivery until successful, and reroute unsuccessful delivery attempts, when necessary.

In summary, the use of QMSG and Radia Messaging Server, as opposed to RADISH, offers performance and reliability enhancements, as well as optional features. For details, see the topics that follow.

Technical Description

The replacement splits the tasks formerly performed by each execution of RADISH among three different components, including the RIM Server:

- QMSG converts data objects to XML and places them in a data directory on the Radia Configuration Server. It doesn't perform table mapping and it doesn't connect to the Radia Integration Server (RIS) as RADISH does.
- Radia Messaging Server picks up data waiting in the data directory and sends it to the RIS Server.
- RIM Server now maps the inventory data to the back-end database.

The overall time it takes to post RIM data to the back-end database may be longer with this solution. The benefit, however, is that the RADISH burden on the Radia Configuration Server is lifted and the use of a local data queue ensures that no data is lost even when there are RIM connection problems. This solution should be considered in cases where the RADISH processing of RIM data presents a performance issue, as well as cases where Radia Configuration Server processing, in general, would benefit from lower overhead. Care should be taken to account for the increased workload placed on the RIS/RIM server.

The following topics are discussed in this chapter to guide you in assessing and implementation the change to your Inventory processing.

- Differences in RIM Information Flow using RADISH versus QMSG with the Radia Messaging Server
- Overview and Alerts!
- Upgrading your RIM Server
- Recommended RIM Server Settings
- Troubleshooting the Radia Messaging Server

Differences in RIM Information Flow using RADISH versus QMSG with the Radia Messaging Server

This section compares the processing flow of posting RIM data using the RADISH executable versus the new Radia Messaging Server and QMSG executable. Review this section to understand which components are involved in making the replacement—for both the Radia Configuration Server machine as well as the RIM Server machine.

Figure 4.1 illustrates the differences in how RIM information is processed on the Radia Configuration Server as it makes its way from the client to the RIM back-end database. Going from left to right, the upper path shows the processing flow using RADISH, while the lower path shows the processing flow using QMSG and the Radia Messaging Server.

Note

Figure 4.1 references the executable sendmsg.exe. SENDMSG was renamed QMSG in May of 2004.

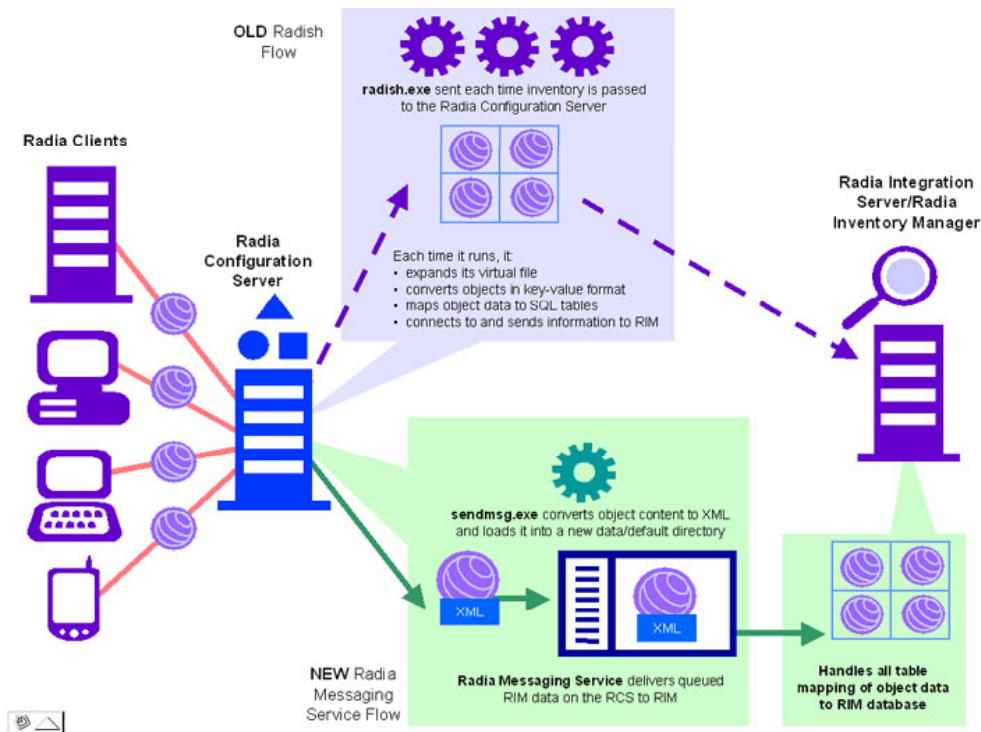


Figure 4.1 ~ RIM data flow from clients to back-end database.

Table 4.1 gives a narrative summary of the alternate RIM processing flows illustrated in Figure 4.1.

Table 4.1 ~ RIM Information Flow Comparisons	
Flow using RADISH	New Flow using Radia Messaging Server with QMSG
1. Client: The client device performs a scan based on the parameters that you gave it using the Radia System Explorer on the Radia Configuration Server. The results are posted in an object for transport to the manager. Object differencing is done on the client.	1. Client: <i>No change.</i>
2. Radia Configuration Server: The Radia Configuration Server receives the objects from the client and processes them according to the parameters that you set in the methods and scripts. <ul style="list-style-type: none"> calls RADISH. 	2. Radia Configuration Server: Same process, but uses a different ZTASKEND method. <ul style="list-style-type: none"> Replacement calls QMSG.
3. RADISH: Posts the information from the Radia Configuration Server directly to the RIS of RIM. Each time it runs, it expands its virtual file, converts objects in key-value format, maps object data to SQL tables, and connects to and then sends information to RIM.	3. A. QMSG: Converts the object's content to XML and loads it into a new data/default directory on the Radia Configuration Server to be picked up by the Radia Messaging Server. <ul style="list-style-type: none"> No table mapping. No connection to RIM. B. Radia Messaging Server: Picks up, routes, and delivers the queued RIM data from the data/default directory on the Radia Configuration Server to RIM and RMP. Configurable using parameters in Table 3.3 on page 57.
4. RIS/RIM: Receives the information from the RADISH executable and inserts it into the database.	4. RIS/RIM: The replacement RIM module receives the information from the Radia Messaging Server. <ul style="list-style-type: none"> Handles all table mapping functions. <i>RIS/RIM Server has a heavier workload; its CPU usage will increase.</i>
5. RIM Database: Receives insertions from RIM, and is queried from the Web page or from customer-managed reporting tools.	5. RIM Database: <i>No change.</i>

Migration Details

Overview and Alerts!

Please read all installation steps before you begin.

- As with all changes involving the Radia Configuration Server, we recommend evaluating the change in a test environment before making changes to your production environment.
- The first step is to install updated and new components on the Radia Configuration Server. You'll be updating your ZTASKEND method, adding a QMSG executable, creating the data directory where QMSG queues data for pickup, and installing the Radia Messaging Server and its nvdkit.tkd module in a new MessagingServer directory.
- The second step is to update your RIM server so that it contains the required rim.tkd module.

ALERTS!

1. Customers who have customized their ZTASKEND need to port their customizations to the new ZTASKEND. See Step 4 of *Installation Steps on the Radia Configuration Server* on page 77.
2. Even after evaluating this change in your test environment, be careful when introducing the change in your production environment. There may be side effects to consider. For details, see *Considering the Impact on Your Production Environment* on page 78.

Table of RCS and RMS Components

Table 4.2 summarizes the files needed for a Radia Messaging Server environment, and where they are placed in your Radia Infrastructure.

Be aware that the Radia Messaging Server folder is created at the same level as the <RCS directory>, not below it.

Table 4.2 ~ Software Needed to Migrate to QMSG/RMS/RIM

Item	Description	Target Location
ZTASKEND	Updated Radia Configuration Server rexx that calls QMSG in place of radish .	Windows: <RCS directory>\rexx UNIX: <RCS directory>/rexx
rms.tkd	New messaging service module that handles queuing, routing, and delivery of RIM data from the Radia Configuration Server to RIM and RMP. This resides in a new Messaging Server directory at the same level as the Radia Configuration Server. For example, C:/Novadigm/MessagingServer.	Windows: C:\Novadigm\MessagingServer UNIX: /opt/Novadigm/MessagingServer
<platform>/nvdkit	The nvdkit executable for running rms.tkd on the Radia Configuration Server. This resides in the new Messaging Server directory at the same level as the Radia Configuration Server. For example, C:/Novadigm/MessagingServer.	Windows: C:\Novadigm\MessagingServer UNIX: /opt/Novadigm/MessagingServer
<platform>/QMSG	New Radia Configuration Server executable that fetches a list of objects, converts them to XML, and places them in a /data/default directory under the Radia Configuration Server directory, to be picked up and delivered by the messaging service module.	Windows: <RCS directory>\bin UNIX: <RCS directory>/exe
rim.tkd	Updated RIM module that handles incoming RIM data from the messaging service module.	Windows: < RIM's -RIS directory> \modules UNIX: < RIM's -RIS directory> /modules

Installation Steps on the Radia Configuration Server

The procedures to install the Radia Messaging Server and update all required RCS components are covered in Chapter 2. See *Creating a Radia Messaging Server Environment on your Radia Configuration Server* on page 18.

Upgrading your RIM Server

Use the procedure below to upgrade an existing RIM Server to Version 4.0, which is provided on the Radia 4 Infrastructure CD.

To upgrade the Radia Inventory Manager Server for use with Radia Messaging Server 2.0

1. Stop the Radia Integration Server [httpd] service.
2. Launch the installation program for the Radia Inventory Manager Server, available from the following platform-specific location on the Radia Infrastructure CD:
 - `\extended_infrastructure\inventory_manager_server\<platform>`
 - For Windows, click on **setup.exe** to launch the installation program.
 - For a UNIX platform, enter the following command:


```
./install
```

 and press **Enter**.
3. Follow the prompts to complete the installation, which upgrades your Radia Inventory Manager Server to the necessary level.
4. If the Radia Integration Service {httpd.tkd} starts automatically after running the installation, stop the service to perform the next step.

Important Note

Normally, taskend.tcl resides in the Radia Configuration Server's **lib** directory. The unpacked location of taskend.tcl is now on the RIM server in the Radia Integration Server's `/etc/rim/lib` folder. RIM will now load all .tcl files from this new directory. This allows customers to place custom scripts there and have them automatically load, avoiding the need to append custom mappings to taskend.tcl.

5. Relocate any *.tcl files in your current Radia Configuration Server **lib** folder, other than **fileaudt.tcl**, to the Radia Integration Server **etc/rim/lib** folder on the RIM Server.

Notes

1. Any customizations made to `taskend.tcl` (or other Radia Configuration Server `.tcl` files) must be relocated to the Radia Integration Server's `/etc/rim/lib` folder, or they will not be loaded.
2. Do not move `fileaudt.tcl` to the RIM server because it is obsolete for RIM as of version 2.1. For more information, read the WARNING lines at the top of `fileaudt.tcl`.

6. Restart the RIS (httpd) service.

Recommended RIM Server Settings

Because of the change in processing tasks, the RIM Server is now handling the majority of the work for posting information to the back-end database. This will most likely increase your RIM Server's CPU usage.

- We recommend that each Radia Configuration Server map to one specific instance of RIM running on its own port and send information through its own dedicated DSN.

You can also increase the 'POLL' setting in the `rms.cfg`. This will slow the amount of information coming to the RIM server. For details, see *Configuring the Poll Interval and Post Quantity* on page 58.

Considering the Impact on Your Production Environment

This change should be implemented carefully in your production environment. Because it allows the Radia Configuration Server to regain CPU cycles previously used by many RADISH processes, you can expect the Radia Configuration Server to process clients faster. As a result of this, some side effects to other infrastructure components may arise due to heavier processing loads. We recommend that you carefully review your Client Connect scheduling so that the rate of client processing is under control, and other parts of your infrastructure do not become overloaded.

For example, policy resolution is one area to consider carefully. As your Radia Configuration Server processes clients faster, the rate of policy resolution increases. In turn, the Radia Policy Server will be making more (concurrent) requests to your corporate LDAP directory. Left unchecked, the corporate directory may not be able to handle this additional load. Thus, you need to keep an eye on your policy resolution so that your corporate LDAP directory is not overloaded.

For additional planning and assistance, please contact HP OpenView Customer Support.

Summary

- Migrating from the use of RADISH to using QMSG with the Radia Messaging Server for processing messages enhances the performance of your Radia Configuration Server and increases the reliability of posting Inventory data.
- The impacts of making the change may include an increase in your RIM Server's CPU usage due to its increased processing role. You may need to adjust RIM Settings and the RMS poll value.
- Migrating to the use of Radia Messaging Server involves changes to your Radia Configuration Server as well as changes to the Radia Integration Server modules.

Troubleshooting

At the end of this chapter, you will:

- Understand how to resolve common error messages in the rms.log file. This log file is located in the Logs directory of the root MessagingServer installation directory.
- Understand how to resolve failed posts.

Troubleshooting the Radia Messaging Server

The Radia Messaging Server log file is located in the Logs directory of the root MessagingServer installation directory.

Problem:

Your Radia Messaging Server log includes WARNING and ERROR messages indicating 'no route defined for default' and 'failed to deliver to default', as shown in Figure 5.1.

```
Warning: router1: To: default, From: <rscs>@<rscs_hostname>, subject: - no route
defined for default
Error: MSG/QUEUE: q2: To: default, From: <rscs>@<rscs_hostname>, subject: -
failed to deliver to default
```

Figure 5.1 ~ Errors when QMSG calls are missing a -to argument.

Solution:

These messages indicate that one or more QMSG calls in ZTASKEND are missing a -to parameter and or value. When this happens, the word 'default' becomes the -to value for that message. Since the Radia Messaging Server does not have a route defined for messages labeled with a -to value of **default**, it cannot route the message and writes these warning and error messages to the log.

The solution is to review the QMSG calls in ZTASKEND and add any missing -to parameters or missing values. As delivered, QMSG -to values include: -to core.rim, -to core.rmp, -to inventory, and -to inventory.wbem, although there may be others. For details, see the topics *QMSG Call Syntax* on page 48 and *Configuring the Register Router Section* on page 60.

Problem:

You receive Error 404 (page not found) or error 500 (server internal error) for all attempted posts to a RIM Server.

Solution:

Review the *.sql files located in the **/etc/sql** folder of your Radia Integration Server. Check the bottom of your *.sql to see if there is a commented out section that looks like:

```
#sql::url . . .
```

The solution for Error 404 or Error 500 is to remove the # (pound sign) to un-comment this line.

- If you do not have any customizations, you can move all of the *.sql files out of the /etc/sql directory (place them in an outside location), and then stop and start RIM. This unpacks the new .sql files, which should fix the error.
- If you have more than one customization, use the following steps to correct the problem and still keep your customizations:
 - a.** Locate any *.sql files that you have customized in the /etc/sql folder, and move them to another, temporary, location.
 - b.** Delete (or move) the remaining *.sql files from the /etc/sql folder.
 - c.** Restart RIM to unpack a new set of *.sql files.
 - d.** Go to where you placed the customized *.sql files, and un-comment the sql::url line at the bottom of each file.
 - e.** Replace the default files that you just unpacked in the /etc/sql folder with the customized files that you edited.
 - f.** Restart the RIM service.

If you have any questions regarding this document or this code, please refer to the HP OpenView support web site.

Summary

- Review the common error messages and solutions given in this topic to troubleshoot Radia Messaging Server problems.
- Failed posts can be the result of the line "sql::init" being commented out in one or more files in the /etc/sql folder of your RIM server's installation directory.



Alternative Radia Messaging Server Configurations

Optional Radia Messaging Server Configurations

The Radia Messaging Server can be adapted to meet various messaging needs. While this Appendix is not comprehensive, it presents a few simple models for using the Radia Messaging Server in alternative configurations. These configurations include:

- *Example 1: Configuring the Radia Messaging Server for Store and Forward* on page 86.
- *Example 2: Configuring Radia Messaging Server to Route to Multiple Queues* on page 89.
- *Example 3: Configuring Radia Messaging Server for Processing Multiple Message Queues* on page 91.

Warning

Examples 2 and 3 are for illustrative purposes. We advise you to contact HP OpenView Technical Support and discuss your individual needs before making substantial changes to your Radia Messaging Server configuration. In addition, remember to fully test any new configurations in a non-production environment, including the use of stress-tests that duplicate production volumes.

Example 1: Configuring the Radia Messaging Server for Store and Forward

The Radia Messaging Server includes store and forward capabilities that allow you to create a multiple-Radia Messaging Server environment. When a Radia Messaging Server is used for store-and-forward processing, messages are forwarded from one Radia Messaging Server to another, before being sent to their final destination. This is illustrated in Figure A.1 below.

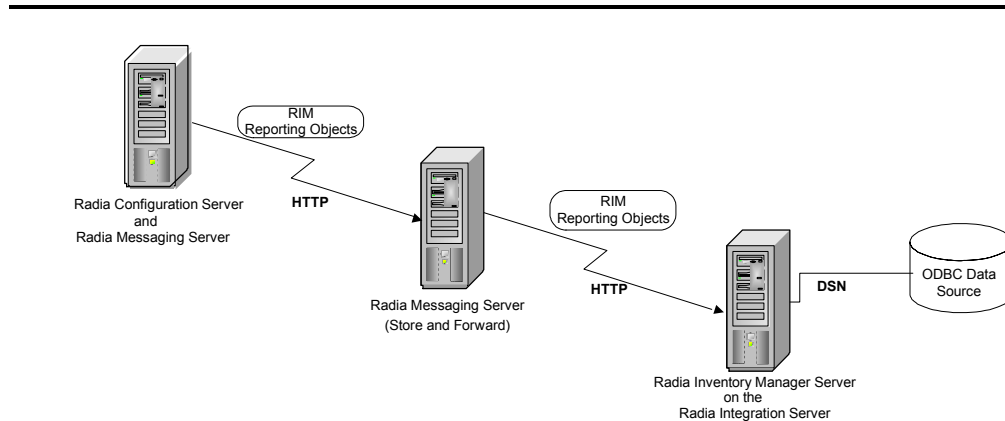


Figure A.1 ~ Radia Messaging Servers configured to pass data using a Store and Forward "Hop".

When forwarding messages from one Radia Messaging Server to another, the first data queue drains very quickly, since there is no data "processing" taking place on the receiving end. For example, you can configure the Radia Messaging Server on the Radia Configuration Server to forward all inventory messages to another, remote, Radia Messaging Server. This configuration drains the inventory messages from the Radia Configuration Server location quickly, freeing up Radia Configuration Server resources for client-resolution tasks.

The following topics explain how to create a store and forward messaging environment:

- *Installing and Configuring a "Receiving" Radia Messaging Server* on page 87.
- *Configuring a Radia Messaging Server to Forward Messages to Another Radia Messaging Server* on page 88.

Installing and Configuring a "Receiving" Radia Messaging Server

A receiving Radia Messaging Server is usually installed remote from the Radia Configuration Server. Messages will be forwarded to this Radia Messaging Server from another Radia Messaging Server. The messages that are received will be placed in the location specified as the **Directory to Scan for Messages** during the installation.

To install a "Receiving" Radia Messaging Service

1. To install a "Receiving" Radia Messaging Server, use the installation procedures discussed in the topic: *Task 4: Install the Radia Messaging Server* on page 21, with the following exception:

When prompted for the **Directory to Scan for Messages** do not accept the default path of `..ConfigurationServer/data/default`. Browse or enter a local directory to scan for messages. If the directory you specify does not exist, it will be created. To maintain consistency across the Radia Messaging Servers in your enterprise, include the `"/data/default"` subdirectory in the named path for the message store location. For example, specify

`C: \<MessageLocation>\data\default` as the location to scan for messages.

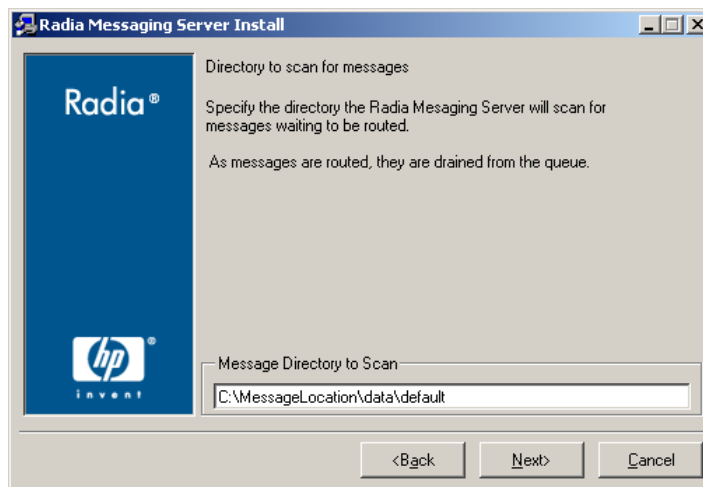


Figure A.2 ~ Specify a local scan directory when installing a "receiving" Radia Messaging Server.

Once started, the receiving Radia Messaging Server deposits messages sent to it in the newly created "Message Directory to Scan" location. The Radia Messaging Server will scan and forward the messages placed in this location to their next destinations, as specified in the `rms.cfg` file.

The log file for the Radia Messaging Server (rms.log) resides in the **Logs** folder of the **MessagingServer** directory. By default, this is C:\Novadigm\MessagingServer\Logs for Windows, or /opt/Novadigm/MessagingServer/Logs for UNIX.

Configuring a Radia Messaging Server to Forward Messages to Another Radia Messaging Server

Use this procedure to modify a previously installed Radia Messaging Server to forward messages to another Radia Messaging Server, instead of to their final destination.

To configure a Radia Messaging Server to Forward Messages to Another Radia Messaging Server

1. Stop the RMS service and edit the rms.cfg file. For details, see *Editing the RMS.CFG File* on page 54.
2. Locate the sections starting with '**msg::register rim**' and '**msg::register rmp**' near the end of the rms.cfg file. Within each section is a URL entry pointing to the host and port of the RIM and RMP servers.
3. For each type of data you want forwarded to another Radia Messaging Server, modify the URL entry. Change the host and port in the URL entry to specify the host and port of the receiving RMS server. Keep the rest of the URL entry the same. The format of the entries are shown below:

For RIM:

```
URL      http://<RMS_hostname:port>/proc/rim/default
```

For RMP:

```
URL      http://<RMS_hostname:port>/proc/xml/obj
```

Where:

RMS_hostname can be specified using an IP address or a DNS name, and

The default *port* number for the Radia Messaging Server is **3461**.

Figure A.3 ~ Edit the URL value in rmp.cfg to specify the host and port of the RMS on page 89 shows where to specify the RMS host and port to forward RIM data to another Radia Messaging Server.

```

msg::register rim {
  TYPE      HTTP
  ADDRESS   {
    PRI      10
    URL      http://111.111.111.11:3461/proc/rim/default
  }
}
msg::register rmp {
  TYPE      HTTP
  ADDRESS   {
    PRI      10
    URL      http://localhost:3466/proc/xml/obj
  }
}

```

 **To forward RIM data to another RMS, specify the receiving RMS host and port in this URL. The RMS port is usually 3461.**

Figure A.3 ~ Edit the URL value in rmp.cfg to specify the host and port of the RMS.

4. Save the changes to rms.cfg.
5. Restart the RMS Service. For details, see *Starting and Stopping the Radia Messaging Server* on page 39.

Example 2: Configuring Radia Messaging Server to Route to Multiple Queues

This example configures the Radia Messaging Server to route messages from a single queue to multiple queues based on their destination. A message's destination is defined by the **-to** parameter value passed from QMSG and stored in the meta-data in the message file.

In the configuration shown in Figure A.4 on page 90, all messages will be placed in the standard location (the directory C:/Novadigm/ConfigurationServer/data/default) when they are created by QMSG. We want to have the Radia Messaging Server route these messages into separate message queues before they are processed. In the code sample that follows, we define the routes for CORE.RIM and CORE.RMP messages to use queue1 and queue2 definitions. The new queue1 and queue2 definitions direct the CORE.RIM messages to the C:/rim directory and the CORE.RMP messages to the C:/rmp directory.

Additional sections must be coded in rms.cfg to complete the routing of the messages. However, this example illustrates the basic concept of routing messages from a single queue to multiple queues, before routing them to processing destinations.

```

msg::register default {
    TYPE      QUEUE

    DIR        ../ConfigurationServer/data/default
    USE        router1

    POLL       10
    COUNT      100
    DELAY      3600
    ATTEMPTS   200
}

msg::register router1 {
    TYPE      ROUTER

    ROUTE     {
        TO     CORE.RIM
        USE    queue1
    }

    ROUTE     {
        TO     CORE.RMP
        USE    queue2
    }
}

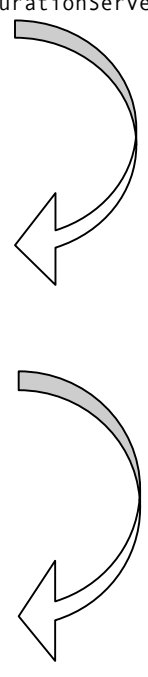
msg::register queue1 {
    TYPE      QUEUE

    DIR       C:/rim
}

msg::register queue2 {
    TYPE      QUEUE

    DIR       C:/rmp
}

```



The diagram illustrates the message flow in the configuration. A curved arrow points from the 'router1' entry in the 'default' queue configuration to the 'router1' configuration block. From the 'router1' block, two curved arrows point to the 'queue1' and 'queue2' configuration blocks, indicating that messages are routed from the router to these specific queues.

Figure A.4 ~ Sample Radia Messaging Server configuration sorting messages into multiple queues.

Additional sections must be coded in rms.cfg to complete the routing of the messages. However, this example shows the basics of how you can route messages from a single queue to multiple queues, before routing to processing destinations.

Example 3: Configuring Radia Messaging Server for Processing Multiple Message Queues

This example configures the Radia Messaging Server to process messages that have been placed by QMSG into multiple data directories for processing. In the example which follows, most messages are sent to and processed from the standard `../ConfigurationServer/data/default` directory, however the PATCH messages are sent by QMSG to a different PATCH queue, whose directory location is defined in RMS.CFG as `../ConfigurationServer/data/queue_rpm`.

Important Note!

This configuration is not recommended for most applications, since it requires modifications to both the QMSG calls in ZTASKEND as well as your Radia Messaging Server configuration. However, there may be instances where this alternative has suitable applications. Please contact an HP OpenView Software Engineer or Technical Support person to discuss a specific implementation.

QMSG and RMS.CFG require the following modifications for using multiple message queues:

- QMSG calls include an initial **-queue** parameter for messages that are to be directed to a processing location other than `../ConfigurationServer/data/default`. (The `../ConfigurationServer/data/default` location is where messages are processed when a '-queue' parameter is not explicitly coded in the QMSG call.)

For example, the following call to QMSG places the objects collected by this call in the queue named **queue_rpm**. The objects collected (PREFACE and ZOBJSTAT) have a Radia Messaging Server '-to' destination of PATCH.

```
'qmsg -queue queue_rpm -to PATCH PREFACE ZOBJSTAT'
```

- The rms.cfg file needs to include a definition for each queue: an 'msg::register default' definition for the messages placed in `/data/default`, as well as an 'msg::register queue_rpm' definition for messages placed in `/data/queue_rpm`. The rms.cfg also needs to include a router definition for messages placed in `/data/queue_rpm` with the TO PATCH destination.

Figures

Figure 1.1 ~ Radia Messaging Server process for routing data from RCS to external locations.	14
Figure 2.1 ~ Radia Messaging Server welcome window.	22
Figure 2.2 ~ End User Licensing Agreement.....	23
Figure 2.3 ~ Radia Messaging Server location window.	24
Figure 2.4 ~ Type the IP Address of the Radia Inventory Manager Server.	25
Figure 2.5 ~ Type the port number of the Radia Inventory Manager Server.	26
Figure 2.6 ~ Type the IP Address of the Radia Management Portal Server.	27
Figure 2.7 ~ Port number for the Radia Management Portal.....	28
Figure 2.8 ~ Store & Forward Port number to receive forwarded messages.	29
Figure 2.9 ~ If connecting to a SQL Database, such as Patch, specify an ODBC Data Set Name. .	30
Figure 2.10 ~ If connecting to a SQL Database, such as Patch, specify the Data Set User Name..	31
Figure 2.11 ~ If connecting to a SQL Database, such as Patch, specify the Password for the DSN User.	32
Figure 2.12 ~ Specify the directory to scan for messages on the Radia Configuration Server.	33
Figure 2.13 ~ Verify the summary.	34
Figure 2.14 ~ The installation is complete.....	35
Figure 2.15 ~ For Patch support, add lines to the Register Router section.	37
Figure 2.16 ~ For Patch Support, add these lines to the end of the rms.cfg file.....	37
Figure 3.1 ~ ZTASKEND invokes QMSG, which places data in a location for Radia Messaging Server processing.	45
Figure 3.2 ~ Sample ZTASKEND code identifying objects collected by QMSG calls, by client phase.....	46
Figure 3.3 ~ ZTASKEND code with QMSG calls to collect larger Inventory objects: Fileaudt, WBEMAUDT, and CLISTATS objects.	48
Figure 3.4 ~ Sample syntax and examples of ZTASKEND calls to QMSG.....	49

Figure 3.5 ~ Messages are sorted in ascending priority, and then processed top down by Radia Messaging Server.	51
Figure 3.6 ~ Edit the QMSG call's -priority parameter to change processing priority by object type.	52
Figure 3.7 ~ Typical rms.cfg, configured to deliver RIM messages and discard RMP messages.	55
Figure 3.8 ~ Edit the DIR value in the 'msg::register default' section of rms.cfg file.	59
Figure 3.9 ~ msg::register router section defines a ROUTE for each QMSG - to destination.	61
Figure 3.10 ~ Set USE as /dev/null to successfully discard certain messages, or drain a queue.	62
Figure 3.11 ~ Enable RMP message processing by changing USE value to rmp.	63
Figure 3.12 ~ Edit the URL value in rmp.cfg to specify the host and port for your RMP server.	63
Figure 3.13 ~ For RMP 1.x, also edit the URL value in rmp.cfg to specify /proc/rmp/default.	64
Figure 3.14 ~ Sample rms.cfg file modified for RIM failover. Code in bold added for failover.	65
Figure 3.15 ~ Log.configure lines added to rms.cfg to limit log size and number.	66
Figure 4.1 ~ RIM data flow from clients to back-end database.	73
Figure 5.1 ~ Errors when QMSG calls are missing a -to argument.	82
Figure A.1 ~ Radia Messaging Servers configured to pass data using a Store and Forward "Hop".	86
Figure A.2 ~ Specify a local scan directory when installing a "receiving" Radia Messaging Server.	87
Figure A.3 ~ Edit the URL value in rmp.cfg to specify the host and port of the RMS.	89
Figure A.4 ~ Sample Radia Messaging Server configuration sorting messages into multiple queues.	90

Tables

Table P.1 ~ Styles	6
Table P.2 ~ Usage.....	6
Table P.3 ~ Terminology*	7
Table 3.1 ~ ZTASKEND calls to QMSG for CORE.RIM and CORE.RMP Data, by Client Connect Phase	47
Table 3.2 ~ ZTASKEND Calls to QMSG by Object Type and Priority	52
Table 3.3 ~ RMS.CFG Parameters used to Define the /Data/Default Queue	57
Table 3.4 ~ Radia Messaging Server Sections for Adding Failover.....	65
Table 4.1 ~ RIM Information Flow Comparisons	74
Table 4.2 ~ Software Needed to Migrate to QMSG/RMS/RIM.....	76

Procedures

To replace the ZTASKEND method	19
To add QMSG.EXE to the Radia Configuration Server	20
To run the install for the Radia Messaging Service	21
To modify/verify the rms.cfg file entries for Patch Manager	36
To encrypt the password entry for the Patch SQL database DSN in rms.cfg.....	38
To modify a QMSG priority assignment	52
To edit the rms.cfg file and parameters	54
To add a WORKERS value to create multiple processes (WORKERS)	59
To disable processing	60
To enable a disabled queue	60
To modify rmp.cfg for routing RMP data	62
To modify rmp.cfg for RMP 1.x.....	64
To set failover in the rms.cfg	65
To upgrade the Radia Inventory Manager Server for use with Radia Messaging Server 2.0	77
To install a "Receiving" Radia Messaging Service	87
To configure a Radia Messaging Server to Forward Messages to Another Radia Messaging Server	88

Index

A

ATTEMPTS value 58

C

customer support 4

D

data/default directory 19

DELAY time 58

DIR value 59

discarding messages 61

draining the message queue 62

E

Error 404 82

ERROR message

 failed to deliver to default 82

EULA window 23

F

failover 25, 64

I

Installation

 Store & Forward Port 28

M

migration to QMSG 70

msg::register 57

N

nvdkit

 encrypt password 38

nvdkit.tkd module 75

P

password

 encrypt 38

Patch

 encrypt password 38

prerequisites 18

PRI value 64

Q

QMSG executable 16, 20

queue

 draining 62

queue, disabling 60

R

Radia Management Portal

 URL for RMP 1.x 64

Radia Messaging Server

 location window 24

 Welcome window 22

Radia Messaging Service

 as a Windows service 15

 starting and stopping 39

RADISH executable 16, 70

RIM server

 settings 78

RIM.TKD 35

rim.tkd module 75

rms.cfg fil

 editing 62

Index

rms.cfg file	15, 25, 37
modifying	54
rms.log	65

S

store and forward	86
system requirements	18

T

technical support	4
-------------------------	---

W

WARNING

no route defined for default	82
Windows service	15
WORKERS parameter	59

Z

ZTASKEND method	19
-----------------------	----