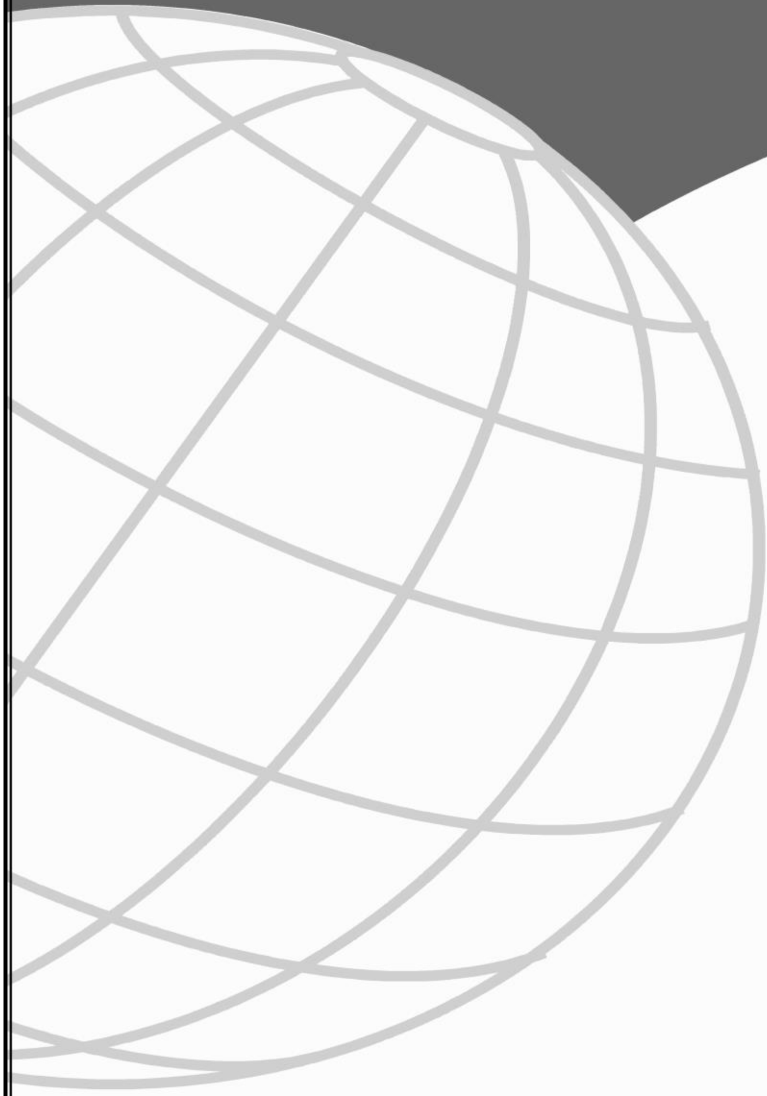


Enabling Service Excellence

Monitoring Messaging Servers

eG Enterprise v6

Restricted Rights Legend

The information contained in this document is confidential and subject to change without notice. No part of this document may be reproduced or disclosed to others without the prior permission of eG Innovations Inc. eG Innovations Inc. makes no warranty of any kind with regard to the software and documentation, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Trademarks

Microsoft Windows, Windows NT, Windows 2003, and Windows 2000 are either registered trademarks or trademarks of Microsoft Corporation in United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Copyright

©2015 eG Innovations Inc. All rights reserved.

Table of Contents

Table of Figures

1

Introduction

Messaging servers play an important role in inter-component communication in an IT infrastructure. With the proliferation of messaging software and its ease of installation and setup, there is the potential for the messaging infrastructure to grow uncontrolled and become difficult to manage. This justifies the need for a monitoring solution that not only reports the current status of the messaging server, but also provides an overview of the health of the messaging infrastructure as a whole.

The eG Enterprise suite provides a bouquet of solutions that specialize in monitoring some of the widely used messaging servers such as MSMQ, WebSphereMQ, FioranoMQ, etc. Unlike silo tools that only provide application-specific monitoring expertise, the eG Enterprise suite provides end-to-end monitoring of an IT infrastructure. This implies that besides messaging servers, eG Enterprise can monitor a wide variety of other applications and network devices that form part of an IT infrastructure. While monitoring a messaging infrastructure for instance, eG Enterprise analyzes the performance of the messaging server in the light of the relationship it shares with other components in the environment, and accordingly determines the state of the server. This approach to monitoring enables eG Enterprise to accurately pinpoint the root-cause of problems that might occur in a messaging infrastructure.

This document focuses on the monitoring capabilities of eG Enterprise with regard to messaging servers.

2

Monitoring MSMQ Servers

Microsoft's MSMQ is a widely used messaging server in Windows environments. In order to ensure that inter-component communication is smooth and swift, the availability and the message processing ability of the MSMQ server should be periodically monitored.

The eG Enterprise suite offers an exclusive *Microsoft MQ* monitoring model (see Figure 2.1), which performs top-down monitoring of the MSMQ server, and proactively alerts administrators to performance issues with the server.

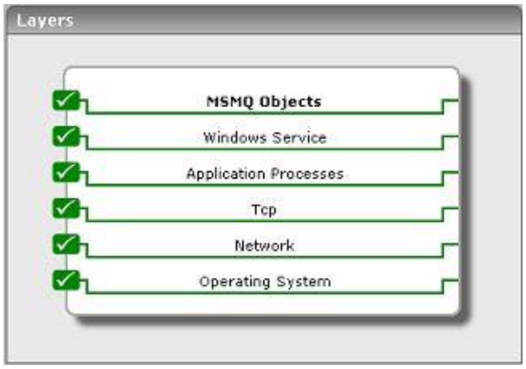


Figure 2.1: Layer model for an MSMQ server

Each layer of Figure 2.1 above is mapped to a set of tests, which when executed on the MSMQ server, extract a variety of performance statistics that reveal the health of the server and the underlying Windows operating system.

The sections to come discuss the **MSMQ Objects** layer alone, as all other layers have already been dealt with in the *Monitoring Unix and Windows Servers* document.

Note:

An eG agent only monitors a maximum of 64 queues.

An eG agent only displays MSMQ traffic coming from and going out to the network.

2.1 The MSMQ Objects Layer

This layer (see Figure 2.2) tracks the health of an MSMQ server.

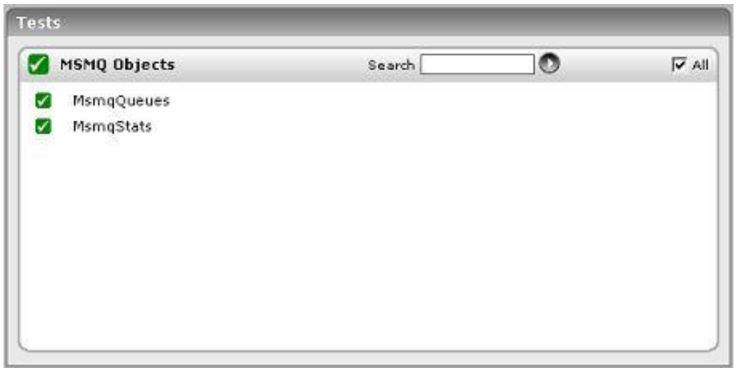


Figure 2.2: Tests mapping to the MSMQ Objects layer

2.1.1 MSMQ Stats Test

This test reports various measures of an MSMQ server.

Purpose	This test measures various usage statistics pertaining to an MSMQ server.		
Target of the test	An MSMQ server		
Agent deploying the test	An internal Agent		
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** – The IP address or host name of the machine where MSMQ has been installed. 3. **PORT** – MSMQ port number (default to 1801)		
Outputs of the test	One set of results for every MSMQ server monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Messages in: This measure indicates the rate of incoming messages handled by the MSMQ service.	Msgs/Sec	An increase or decrease in incoming message rate can represent a change in system load. Alternately, a sudden decrease in incoming message rate can indicate that due to some reason the MSMQ server may be unable to receive messages.
	Messages out: The rate of outgoing messages handled by the MSMQ service.	Msgs/Sec	An increase or decrease in outgoing message rate can represent a change in system load. Alternately, a sudden decrease in outgoing message rate can indicate that the MSMQ server is unable to send messages.

	Current sessions: The total number of open network sessions to MSMQ.	Number	Consistent high values indicate that either there is some persistent activity on the MSMQ server. On the other hand, a high value of current sessions could also mean that the MSMQ server is not able to satisfy user requests and release user sessions. Correlation of this metric with other MSMQ metrics is essential to identify the exact source of the problem.
	Error messages: The total number of accesses that resulted in an error reply.	Number	There could be several reasons for errors occurring during access to an MSMQ server. Possible reasons include: 1. Errors occurring while accessing the public queue 2. Non-existence of the queues being accessed 3. Permission problems for the type of access being requested.
	Total messages: The total number of messages that are waiting to be received locally or to be sent to other destinations.	Number	If the cumulative size of all the messages exceeds 2GB then the newer messages may not be received/sent until some messages are cleared.
	Growth rate: This measure indicates the growth of messages per second in the message queue.	Msgs/Sec	An increase or decrease in growth rate can represent a change in traffic patterns handled by the MSMQ server.

Note:

The measure *Error messages* is not available in the Windows 2000 version of the product.

2.1.2 MSMQ Queues Test

The MsmqQueues test monitors the message queues on an MSMQ server and reports key statistics pertaining to each of the queues.

Purpose	Monitors the message queues on an MSMQ server and reports key statistics pertaining to each of the queues
Target of the test	An MSMQ server

<table>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal Agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> – The IP address or host name of the machine where MSMQ has been installed.
3. <b>PORT</b> – MSMQ port number (default to 1801)</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every queue on the MSMQ server monitored</td></tr>
<tr><td rowspan="5"><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td><b>Traffic in journal queue:</b>
Indicates the total number of bytes that reside in the journal queue.</td><td>KB</td><td></td></tr>
<tr><td><b>Traffic in queue:</b>
Indicates the total number of bytes that reside in the MSMQ queue.</td><td>KB</td><td></td></tr>
<tr><td><b>Messsages in journal queue:</b>
Indicates the number of messages inside a journal queue.</td><td>Number</td><td>Journal queues allow you to save copies of messages as they are processed. You can store copies of the messages you send from your computer in the local machine journal queue, or you can store copies of messages removed from a queue in that queue's journal on the server.
Journal storage can be useful if you need to resend a message at a later point.
Journal queues have a maximum size, called a quota. This size is limited by the disk capacity of the computer on which the queue resides. If the value of this measure reaches the quota set for the journal queue, then the messages that would have been sent to the journal queue are no longer saved in the system. No errors are raised when this occurs. It is important that you periodically purge your journal queues to prevent this from occurring.</td></tr>
<tr><td><b>Messages in queue:</b>
Indicates the number of messages inside a queue.</td><td>Number</td><td>A very high value indicates that the MSMQ server is unable to process the messages faster. A value zero indicates that no messages are in queue to process.</td></tr>
</table>

3

Monitoring WebSphere MQ Servers

The WebSphere MQ server enables messages to be exchanged, either synchronously or asynchronously, between application programs running on one or more target systems. The three most critical components of the server are:

- The queue manager
- The message queue
- The channel

A queue manager is a program that provides messaging services to applications. The queue manager ensures that messages are sent to the correct queue or are routed to another queue manager. The queue manager processes both the MQI calls that are issued to it, and the commands that are submitted to it (from whatever source). The queue manager generates the appropriate completion codes for each call or command.

A queue is a container for messages. Messages can be retrieved from, or added to, the queue, one at a time, by applications that are connected to the queue manager that owns the queue.

Channels are of two types. A **message channel** is a unidirectional communications link between two queue managers that is used to transfer messages between them. An **MQI channel** is bi-directional and connects an application (MQI client) to a queue manager on a server machine for the transfer of MQI calls and responses.

Figure 3.1 depicts the architecture of a WebSphere MQ server:

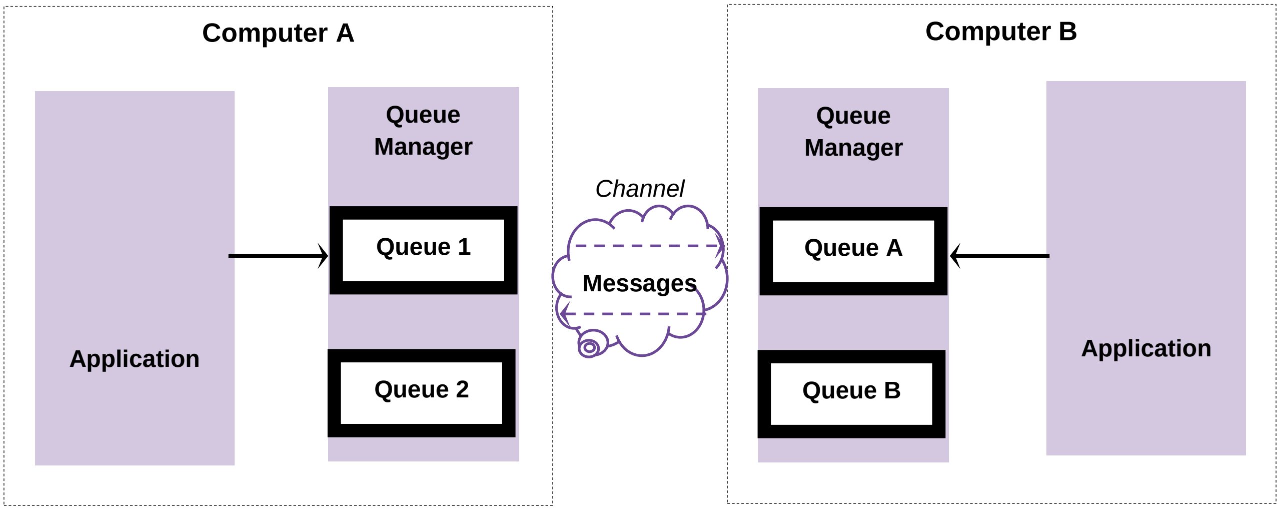


Figure 3.1: Architecture of an IBM MQ server

Disruption in the communication between two components in a WebSphere MQ infrastructure could be caused by the non-availability of the WebSphereMQ server, difficulties in connecting to the queue manager, or due to the queue manager's inability to process data quickly. For communication to resume, administrators should first figure out where the real problem lies. This can be achieved only if the WebSphereMQ server's components are continuously monitored, and potential communication problems accurately detected.

eG Enterprise prescribes a specialized *WebSphere MQ* monitoring model (see Figure 3.2) that monitors the performance of the core WebSphere MQ components such as the queue manager, the message queues, and the channels, proactively identifies slowdowns, and alerts administrators to the anomaly before any loss of communication occurs.

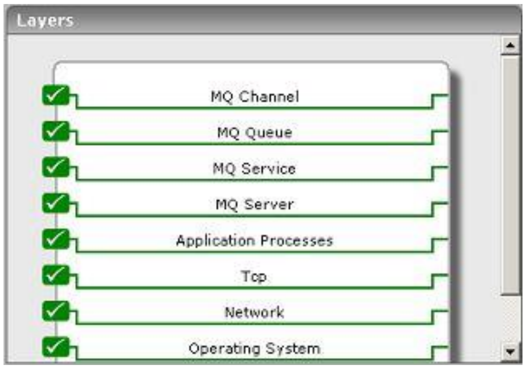


Figure 3.2: Layer model of a WebSphere MQ server

Each layer of Figure 3.2 is mapped to a wide range of tests, which execute on the WebSphere MQ server and collect performance statistics that reveal the following:

- Is the WebSphere MQ server available? Are clients able to connect to the server?
- Are any of the queue managers unavailable?
- Is there a delay in connecting to any queue manager?

- Have too many messages been enqueued?
- Are the queues processing messages too slowly?
- Is the local queue full?
- Are clients able to connect to the channels?
- How is the load on the channels?
- When was the last time the channel transmitted a message? Has it been too long since then?

The sections that follow elaborate on the tests mapped to the top 4 layers of Figure 3.2. For details on the other layers, please refer to the *Monitoring Unix and Windows Servers* document.

3.1 Pre-Requisites for Monitoring IBM WebSphere MQ Servers

1. For monitoring IBM WebSphere MQ server version 6 and below, the following jar files need to be copied from the **[WebSphere MQ install directory/java/lib]** directory to the **<EG_INSTALL_DIR>/lib** directory:
   - com.ibm.mq.jar
   - connector.jar
   - com.ibm.mq.pcf.jar
2. For monitoring WebSphere MQ version 7.0, the following jar files need to be copied to the **<EG_INSTALL_DIR>/lib** directory:
   - com.ibm.mq.jar
   - com.ibm.mq.jmqi.jar
   - com.ibm.mq.headers.jar
   - com.ibm.mq.pcf.jar
   - com.ibm.mq.commonservices.jar
   - connector.jar

   After copying the jar files, remember to restart the eG agent. If MQ monitoring is done in an agentless manner, these jar files should be available on the remote agent that will perform the monitoring.
3. Configure the eG agent to monitor the queue manager and its queues and topics.

The section that follows elaborately discusses how to fulfill pre-requisite number 3 above.

3.1.1 Configuring the eG Agent to Monitor the Queue Manager and its Queues and Topics

Typically, to monitor a queue manager, the eG agent needs to be configured with access to the default server connection channel on WebSphere MQ – i.e., the **SYSTEM.DEF.SVRCONN** channel. In high security environments however, administrators may prefer not to expose this default channel and its access privileges to the eG agent. In such environments therefore, you need to do the following to enable the eG agent to monitor the queue manager:

1. Create a Custom Channel
2. Create a User Group

3. Grant the user group access privileges to the queue manager, queues and topics
4. Assign the user group to the custom channel
5. Configure the eG tests to use the custom channel and the credentials of the user to monitor queues and topics

Let us now discuss each of the steps mentioned above in the following pages.

3.1.1.1 Creating a Custom Channel

To create a custom channel for monitoring a Queue manager, do the following:

Execute the following command from the bin directory of the MQ Install directory to start the MQ Series commands (MQSC) for the queue manager:

runmqsc <Queue Manager name>

For example, if the name of the Queue manager is **eGQMGR**, then the command will be as follows:

runmqsc eGQMGR

Once the MQSC is started, you can execute commands to create a custom channel. Say for example, the name of the custom channel is **eGChanne**l, execute the commands in the command prompt as follows:

DEFINE CHANNEL<eGChannel> CHLTYPE<SVRCONN> TRPTYPE<TCP>

If you wish to add a description to the custom channel, then you can do so by adding a DESCR section to the above mentioned command. In our example, the description is **Server-Connection channel for eG**. Therefore the command that needs to be executed is as follows:

DEFINE CHANNEL<eGChannel> CHLTYPE<SVRCONN> TRPTYPE<TCP> DESCR<Server-Connection channel for eG >

If the command exceeds the stipulated number of characters that need to be typed in the command prompt, then the command can be split by a '+' symbol. In such cases, the command can be executed as follows (see Figure 1):

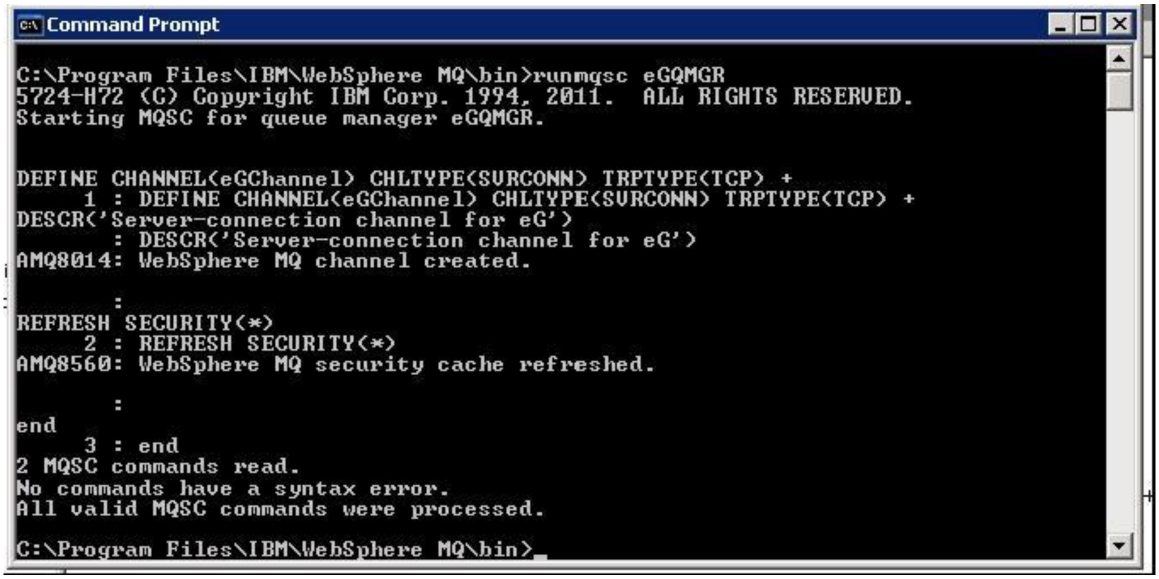
DEFINE CHANNEL<eGChannel> CHLTYPE<SVRCONN> TRPTYPE<TCP> +

DESCR<Server-Connection channel for eG >

The custom WebSphere MQ channel will now be created. For a custom channel to take effect in the WebSphere MQ server, either the security cache or the queue manager of the server needs to be refreshed. If the queue manager is refreshed, the current messages of the server may get deleted. Therefore, refresh the security cache by issuing the following command (see Figure 1):

REFRESH SECURITY<*>

Once the security cache is refreshed successfully, you can stop the MQ Series commands by issuing the following command:



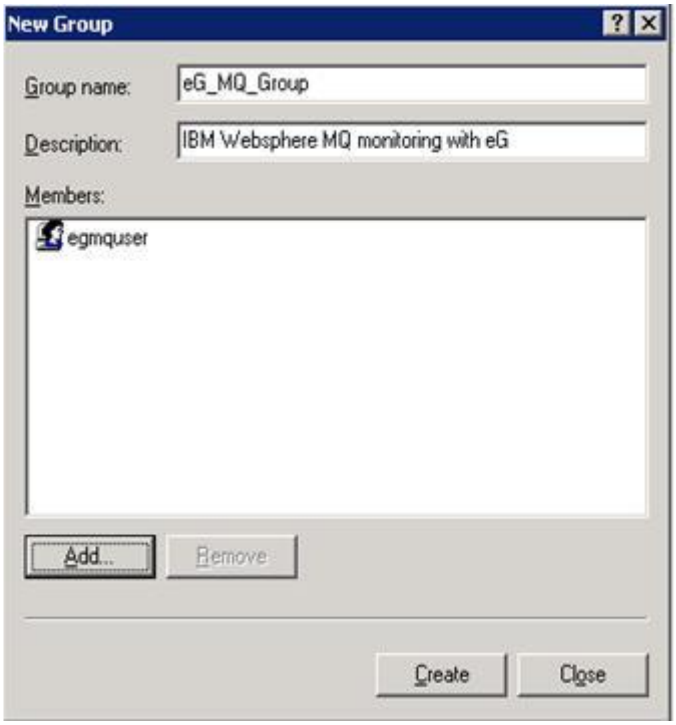
```
C:\Program Files\IBM\WebSphere MQ\bin>runmqsc eGQMGR
5724-H72 (C) Copyright IBM Corp. 1994, 2011.  ALL RIGHTS RESERVED.
Starting MQSC for queue manager eGQMGR.


DEFINE CHANNEL(eGChannel) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
     1 : DEFINE CHANNEL(eGChannel) CHLTYPE(SVRCONN) TRPTYPE(TCP) +
DESCR('Server-connection channel for eG')
       : DESCR('Server-connection channel for eG')
AMQ8014: WebSphere MQ channel created.

       :
REFRESH SECURITY(*)
     2 : REFRESH SECURITY(*)
AMQ8560: WebSphere MQ security cache refreshed.

       :
end
     3 : end
2 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.

C:\Program Files\IBM\WebSphere MQ\bin>_
```

Figure 3.3: Creating a custom channel

The custom channel is now ready for monitoring WebSphere MQ server.

3.1.1.2 Creating a User Group

In order to monitor the WebSphere MQ server, a user group with at least one user is required. The user group may be any operating system user group and a user may be a domain user in case of Windows platforms or a Unix user in case of Unix platforms. You can also associate an existing user to a new user group - for e.g., **eG_MQ_Group** as shown in Figure 3.4.



Group name: eG_MQ_Group

Description: IBM Websphere MQ monitoring with eG

Members:

egmquser

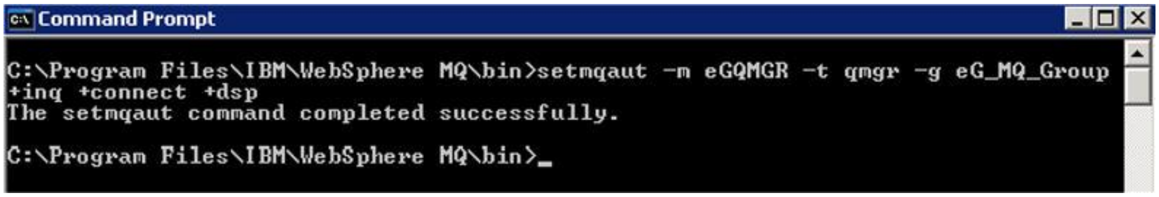
Figure 3.4: Creating a new user group

The user group thus created is to be assigned to the newly created custom channel so that the WebSphere MQ server can be monitored.

3.1.1.3 Enabling the User Group to Access the Queue Manager and its Queues and Topics

In order to associate the user group with the custom channel, you need to grant permission to the user group to access the queue manager and its corresponding queues. Execute the following command from the bin directory of the MQ Install directory (see Figure 3) to grant permission to the user group to access the queue manager.

setmqaut -m eGQMGR -t qmgr -g eG_MQ_Group +inq +connect +dsp

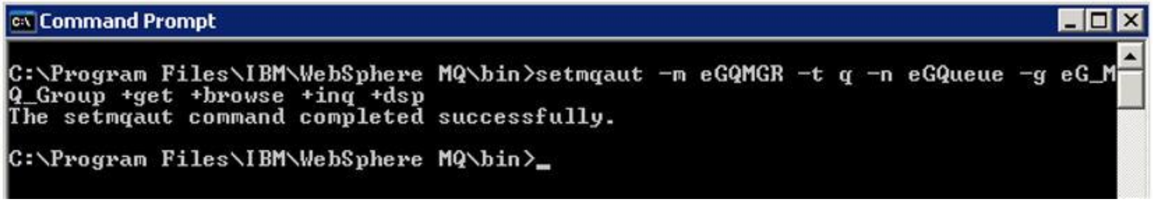


```
C:\Program Files\IBM\WebSphere MQ\bin>setmqaut -m eGQMGR -t qmgr -g eG_MQ_Group
+inq +connect +dsp
The setmqaut command completed successfully.

C:\Program Files\IBM\WebSphere MQ\bin>_
```

Figure 3.5: Granting permissions to the user group to access the custom channel

Once the command is executed successfully, you will need to grant permission to the user group to access the queues of the queue manager. The queues may be either user-defined or pre-defined system queues. Execute the following command to grant permission to the user group to access a user-defined queue named eGQueue:

setmqaut -m eGQMGR -t q -n eGQueue -g eG_MQ_Group +get +browse +inq +dsp



```
C:\Program Files\IBM\WebSphere MQ\bin>setmqaut -m eGQMGR -t q -n eGQueue -g eG_M
Q_Group +get +browse +inq +dsp
The setmqaut command completed successfully.

C:\Program Files\IBM\WebSphere MQ\bin>_
```

Figure 3.6: Granting permission to the user group to access the user-defined queues

Execute the commands as mentioned below to grant permission for the user group to access the pre-defined system queues of your choice.

setmqaut -m eGQMGR -t q -n MQM.SYS.DEAD.QUEUE -g eG_MQ_Group +get +browse +put +inq +dsp +passall

setmqaut -m eGQMGR -t q -n SYSTEM.BROKER.AUTH* -g eG_MQ_Group +get +put +inq +set +dsp

setmqaut -m eGQMGR -t q -n SYSTEM.*.*.*.* -g eG_MQ_Group +inq +dsp

setmqaut -m eGQMGR -t q -n SYSTEM.BROKER.DEPLOY.* -g eG_MQ_Group +get +put +inq +dsp

setmqaut -m eGQMGR -t q -n SYSTEM.*.* -g eG_MQ_Group +inq +dsp

setmqaut -m eGQMGR -t q -n SYSTEM.*.*.* -g eG_MQ_Group +inq +dsp

setmqaut -m eGQMGR -t q -n SYSTEM.MQEXPLORER.*.* -g eG_MQ_Group +get +put +inq +set +dsp

Note

The **MQ GET CALL** permission has to be granted to the user group, so that it can do the following:

- To enable the **WebSphere MQ Local Queues** test to collect detailed diagnostics
- To enable the **WebSpere MQ Queue Details** test to report metrics related to messages and to collect detailed diagnostics.

If you do not want to assign this permission to any user group that will be used by the eG agent for monitoring queues, you can disable the tests mentioned above.

Execute the following command to grant permission for the user group to access the topics of the queue manager:

```
setmqaut -m eGQMGR  -t topic -n "**" -g  eG_MQ_Group +dsp +ctrl
```

3.2 The MQ Server Layer

Using the **WebSphereMQ** test associated with this layer, administrators can verify whether or not the WebSphere MQ server is available.

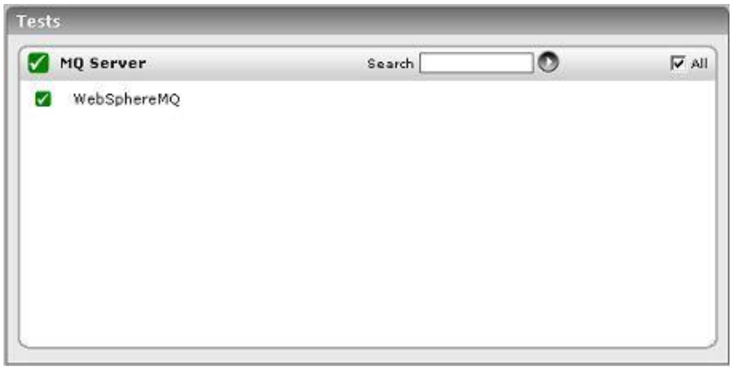


Figure 3.7: The tests associated with the MQ Server layer

3.2.1 WebSphereMQ Test

This test checks the availability of a WebSphere MQ server to handle connection requests from clients.

Purpose	Checks the availability of a WebSphere MQ server to handle connection requests from clients
Target of the test	A WebSphere MQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **USER** – If you want to login as a specific MQ user to execute this test, then specify a valid user name in the **USER** text box. The test will fail if an invalid user name is specified here. If no such authentication is required, then this parameter can be set to 'none'. 5. **PASSWORD** - If a specific **USER** is entered, then the password of that user has to be specified in the **PASSWORD** text box. 6. **CONFIRM PASSWORD** - Confirm the password by retyping it in the **CONFIRM PASSWORD** text box. 7. **SERVERCONNCHANNEL** - The name of the server connection channel for the WebSphere MQ server. The default value is "SYSTEM.DEF.SVRCONN". If you prefer not to use this default server connection channel for monitoring purposes, then you can create a custom channel in the WebSphere MQ server and use it for monitoring. The steps for achieving this have been detailed in Section 3.1.1.1 above.		
Outputs of the test	One set of results for every WebSphere MQ server monitored		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	MQ port availability: Reflects whether the port on which the WebSphere MQ server is listening is available for connections	Percent	This metric takes a value of 100 if the port is available, and 0 if it is not.
	MQ connection availability: Indicates whether connection to the MQ server is available for MQ clients or not	Percent	This metric is 100 if the eG agent is able to connect to the MQ server using the parameters specified (e.g., serverConnChannel). The value of this metric is 0 if the agent is not able to connect to the MQ server. Connection problems can occur if the server is down or unavailable. Invalid user name/password can also result in connection failures.

3.3 The MQ Service Layer

Client applications must connect to the queue manager of the WebSphere MQ server for transmitting messages. If the queue manager is unavailable, inter-component traffic could come to a halt! The test associated with the **MQ Service** layer (see Figure 3.8) periodically checks whether the queue manager is available or not, and alerts administrators of its state, so that the impending communication loss can be averted. The test also reports on the time taken to connect to the queue manager, so that potential connection slowdowns can be promptly identified.

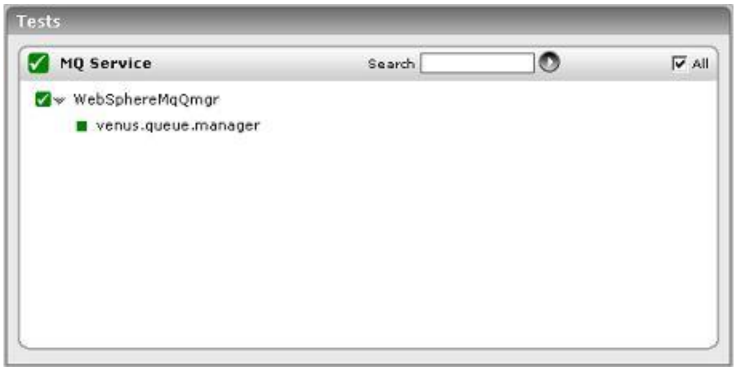


Figure 3.8: Test associated with the MQ Service layer

3.3.1 WebSphereMqQmgr Test

This test emulates a client connecting to a WebSphere MQ queue manager and checks its availability and responsiveness.

Purpose	Checks the availability and responsiveness of a WebSphere MQ queue manager		
Target of the test	A WebSphere MQ server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **USER** – If you want to login as a specific MQ user to execute this test, then specify a valid user name in the **USER** text box. The test will fail if an invalid user name is specified here. If no such authentication is required, then this parameter can be set to 'none'. 5. **PASSWORD** - If a specific **USER** is entered, then the password of that user has to be specified in the **PASSWORD** text box. 6. **CONFIRM PASSWORD** - Confirm the password by retyping it in the **CONFIRM PASSWORD** text box. 7. **SERVERCONNCHANNEL** - The name of the server connection channel for the WebSphere MQ server. The default value is "SYSTEM.DEF.SVRCONN". If you prefer not to use this default server connection channel for monitoring purposes, then you can create a custom channel in the WebSphere MQ server and use it for monitoring. The steps for achieving this have been detailed in Section 3.1.1.1 above.		
Outputs of the test	One set of results for every WebSphere MQ queue manager in a WebSphere MQ server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Queue manager availability: This metric reflects the availability of the queue manager.	Percent	Availability is 100 if the queue manager is servicing connections, and is 0 if the connection to the queue manager fails.

	Connection time to queue manager: This metric reflects the time taken for a connection to the queue manager.	Secs	

3.4 The MQ QUEUE Layer

The layer monitors the message queues of an MQ server. The test associated with this layer is:

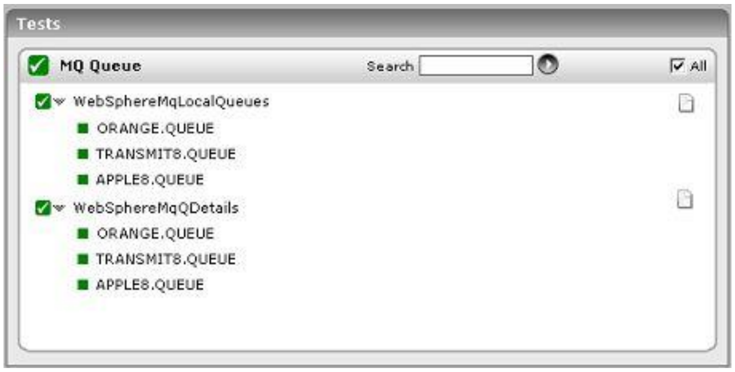


Figure 3.9: Test associated with the MQ Queue layer

3.4.1 WebSphereMqQDetails Test

This test connects to a WebSphere MQ server, auto-discovers the local queues that have been configured for this server, and provides detailed reports on the messages being added to and removed from each of the local queues.

Purpose	Provides detailed reports on the messages being added to and removed from each of the local queues
Target of the test	A WebSphere MQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **USER** – If you want to login as a specific MQ user to execute this test, then specify a valid user name in the **USER** text box. The test will fail if an invalid user name is specified here. If no such authentication is required, then this parameter can be set to 'none'. 5. **PASSWORD** - If a specific **USER** is entered, then the password of that user has to be specified in the **PASSWORD** text box. 6. **CONFIRM PASSWORD** - Confirm the password by retyping it in the **CONFIRM PASSWORD** text box. 7. **SERVERCONNCHANNEL** - The name of the server connection channel for the WebSphere MQ server. The default value is "SYSTEM.DEF.SVRCONN". If you prefer not to use this default server connection channel for monitoring purposes, then you can create a custom channel in the WebSphere MQ server and use it for monitoring. The steps for achieving this have been detailed in Section 3.1.1.1 above. 8. **IGNORESYSTEMQUEUES** - If SYSTEM queues are to be monitored, then set the **IGNORESYSTEMQUEUES** flag to **No**. If not, set the flag to **Yes**. 9. **INCLUDE QUEUES** - Specify a comma-separated list of queue names or queue name patterns to be monitored. For example, your specification can be: VIDEO_*,*.NDURABLE.*,*_TDS_IN. In this case, the test will monitor only those queues with names that begin with "VIDEO_", names that contain the string ".NDURABLE.", and names that end with "_TDS_IN". **Note:** If a SYSTEM queue or a pattern that matches a SYSTEM queue is configured in the **INCLUDE QUEUES** text box, then this test will monitor such SYSTEM queues, even if the **IGNORESYSTEMQUEUES** flag is set to **Yes**. 10. **DELAYLIMIT** - Denotes the time limit beyond which the eG agent regards a message as a delayed message. For e.g., if the delayLimit is 300, the eG agent will regard all messages that have remained in the queue for 5 mins or more as a delayed message. 11. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise system embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option against **DETAILED DIAGNOSIS**. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.
Outputs of the test	One set of results for every local queue configured for the WebSphere MQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	New messages added: The number of new messages added to the queue	Number	To compute this measure, the agent remembers the messages in the queue at the end of a measurement period, and then compares the messages that are in the queue during the next measurement period.
	Old messages removed: The number of old messages removed from the queue during the last measurement period	Number	To track this metric, the agent tracks the messages in the queue at any instant of time. If a message is not available when the agent checks the queue again, the message is added to the count of old messages removed.
	Delayed messages: The number of messages currently in the queue that have been lying in the queue for more than the delayLimit	Number	The time when the message was put into the queue is used to determine how long the message has been delayed. If this test is being executed from a remote agent (i.e., not the local agent), it is essential that the system time of the remote agent and the system that hosts the MQ server are synchronized. Detailed diagnosis for this measure provides details about which messages have been delayed. Information such as the PutDate (when the message was added to a queue), the message ID, the application that put the message, the sequence number of the message, the OS user who is running the application, and the length of the message is reported as part of the detailed diagnosis for this test.
	Avg time in queue: The average time that a message has remained in the queue	Secs	This metric is computed by determining how many messages are in the queue, and determining the time that each message has been in the queue.
	Max time in queue: The maximum time that any of the current messages has spent in the queue.	Secs	This value can give an indication if any message is not being removed from the queue for an unusually long period.

3.4.2 WebSphereMqLocalQueues Test

This test connects to a WebSphere MQ server, auto-discovers the local queues that have been configured for this server, and monitors the status of each of the local queues.

Purpose	Monitors the status of each of the local queues
Target of the test	A WebSphere MQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **USER** – If you want to login as a specific MQ user to execute this test, then specify a valid user name in the **USER** text box. The test will fail if an invalid user name is specified here. If no such authentication is required, then this parameter can be set to 'none'. 5. **PASSWORD** - If a specific **USER** is entered, then the password of that user has to be specified in the **PASSWORD** text box. 6. **CONFIRM PASSWORD** - Confirm the password by retyping it in the **CONFIRM PASSWORD** text box. 7. **SERVERCONNCHANNEL** - The name of the server connection channel for the WebSphere MQ server. The default value is "SYSTEM.DEF.SVRCONN". If you prefer not to use this default server connection channel for monitoring purposes, then you can create a custom channel in the WebSphere MQ server and use it for monitoring. The steps for achieving this have been detailed in Section 3.1.1.1 above. 8. **IGNORESYSTEMQUEUES** - If SYSTEM queues are to be monitored, then set the **IGNORESYSTEMQUEUES** flag to **No**. If not, set the flag to **Yes**. 12. **INCLUDE QUEUES** - Specify a comma-separated list of queue names or queue name patterns to be monitored. For example, your specification can be: VIDEO_*,*.NDURABLE.*,*_TDS_IN. In this case, the test will monitor only those queues with names that begin with "VIDEO_", names that contain the string ".NDURABLE.", and names that end with "_TDS_IN". **Note:** If a SYSTEM queue or a pattern that matches a SYSTEM queue is configured in the **INCLUDE QUEUES** text box, then this test will monitor such SYSTEM queues, even if the **IGNORESYSTEMQUEUES** flag is set to **Yes**. 9. **IGNOREQUEUES** - Takes a default value of "none". If so, none of the queues (other than the SYSTEM queues) are ignored while monitoring an MQ server. Otherwise, this parameter represents a comma separated list of queue names or queue name patterns. Any queue name that matches one of the patterns specified for this test is not considered for monitoring by the eG agent. For example, your specification can be:*.MQSC.*,*_XML_IN. In this case, the test will not monitor those queues with names that begin with that contain the string ".MQSC.", and names that end with "_XML_IN". 10. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise system embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option against **DETAILED DIAGNOSIS**. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: ▪ The eG manager license should allow the detailed diagnosis capability ▪ Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Outputs of the test	One set of results for every local queue configured for the WebSphere MQ server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Current queue depth: Reports the current number of messages in the local queue.	Number	The detailed diagnosis this measure provides information on the messages in a queue. Using the detailed diagnosis, an operator can determine which applications are currently putting messages into the queue, the number of messages for each application that are in the queue, the average time that a message from an application spends in the queue, and the size of messages in the queue distributed across applications accessing the queue.
	Max queue depth: Reports the maximum number of messages that can be held at any instant of time in the local queue.	Secs	
	Messages in queue: Reflects the current percentage occupancy of the queue, and is computed as the ratio of the current queue depth to the maximum queue depth.	Percent	Since messages may be lost when the queue occupancy exceeds 100%, this value should be less than 100% at all times.
	Open inputs: The current number of handles that are currently valid for removing messages from the queue	Number	An unusually large number of input handles could imply either an unexpectedly large number of readers.
	Open outputs: The current number of handles that are currently valid for adding messages to the queue.	Number	

<table>
<tr>
<td></td>
<td><b>Message get status:</b>
Indicates whether/not get operations are allowed for this queue.</td>
<td></td>
<td>The values that this measure can report and their corresponding numeric values are listed in the table below:
<table>
<tr><th>Measure Value</th><th>Numeric Value</th></tr>
<tr><td>Allowed</td><td>0</td></tr>
<tr><td>Inhibited</td><td>1</td></tr>
</table>
<b>Note:</b>
By default, this measure reports the <b>Measure Value</b>s listed in the table above to indicate whether/not get operations are allowed for a queue. In the graph of this measure however, the same is represented using the numeric equivalents only.</td>
</tr>
<tr>
<td></td>
<td><b>Message put status:</b>
Indicates whether/not put operations are allowed for this queue.</td>
<td></td>
<td>The values that this measure can report and their corresponding numeric values are listed in the table below:
<table>
<tr><th>Measure Value</th><th>Numeric Value</th></tr>
<tr><td>Allowed</td><td>0</td></tr>
<tr><td>Inhibited</td><td>1</td></tr>
</table>
<b>Note:</b>
By default, this measure reports the <b>Measure Value</b>s listed in the table above to indicate whether/not put operations are allowed for a queue. In the graph of this measure however, the same is represented using the numeric equivalents only.</td>
</tr>
</table>

3.5 The MQ Channel Layer

This layer tracks the health of channels (see Figure 3.10). The tests associated with this layer are:

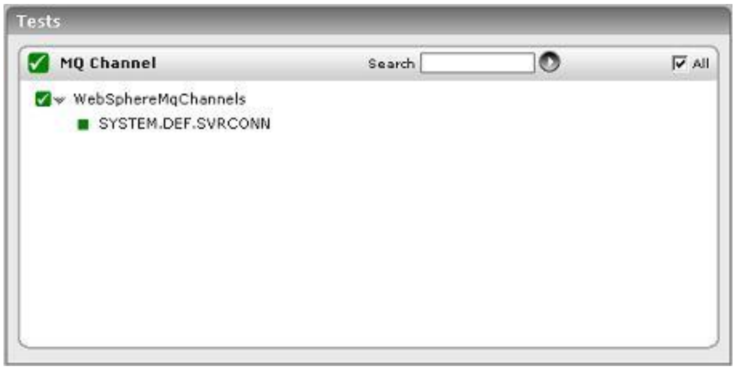


Figure 3.10: Tests associated with the MQ Channel layer

3.5.1 WebSphereMqChannels Test

This test connects to a WebSphere MQ server, auto-discovers the channels that have been configured for this server, and monitors the status of each of these channels.

Purpose	Monitors the status of each of the channels
Target of the test	A WebSphere MQ server
Agent deploying the test	An internal agent
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **USER** – If you want to login as a specific MQ user to execute this test, then specify a valid user name in the **USER** text box. The test will fail if an invalid user name is specified here. If no such authentication is required, then this parameter can be set to 'none'. 5. **PASSWORD** - If a specific **USER** is entered, then the password of that user has to be specified in the **PASSWORD** text box. 6. **CONFIRM PASSWORD** - Confirm the password by retyping it in the **CONFIRM PASSWORD** text box. 7. **SERVERCONNCHANNEL** - The name of the server connection channel for the WebSphere MQ server. The default value is "SYSTEM.DEF.SVRCONN". If you prefer not to use this default server connection channel for monitoring purposes, then you can create a custom channel in the WebSphere MQ server and use it for monitoring. The steps for achieving this have been detailed in Section 3.1.1.1 above. 8. **IGNORESYSTEMCHANNELS** - If SYSTEM channels are to be monitored, then set this flag to **No**. If not, set the flag to **Yes**. 9. **INCLUDE CHANNELS** - Specify a comma-separated list of channel names or channel name patterns to be monitored. For example, your specification can be: _*.EG.*,CLNT.JPM.*,*.CSI.SVRCONN_. In this case, the test will monitor only those channels with names that embed the string _".EG."_, names that start with _"CLNT.JPM"_, and names that end with _".CSI.SVRCONN"_. **Note:** If a SYSTEM channel or a pattern that matches a SYSTEM channel is configured in the **INCLUDE CHANNELS** text box, then this test will monitor such SYSTEM channels, even if the **IGNORESYSTEMCHANNELS** flag is set to **Yes**. 10. **IGNORECHANNELS** - Takes a default value of "none". If so, none of the channels (other than the SYSTEM channels) are ignored while monitoring an MQ server. Otherwise, this parameter represents a comma separated list of pattern of pattern names that should be excluded from monitoring. Any channel name that matches one of the patterns specified for this test will not considered for monitoring by the eG agent. For example, your specification can be: _*.DTC.*,CUST.CIT.*_. In this case, the test will not monitor all those channels with names that embed the string _".DTC."_ and names that begin with _"CUST.CIT."_.
Outputs of the test	One set of results for every channel configured for the WebSphere MQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	MQ channel availability: Reports the current availability of a channel.	Percent	A value of 100% signifies that the channel is available for clients to connect to. A value of 0% indicates channel unavailability.
	Current messages: The number of messages being currently handled by the channel.	Number	A continuous increase of this value signifies that the receiver is not able to cope up with the incoming traffic.
	Messages handled: The number of messages that transmitted through the channel during the last measurement period	Number	
	Buffers received: The number of buffers of data received over the channel during the last measurement period.	Number	This metric indicates the workload over a channel.
	Buffers sent: Indicates the number of buffers of data transmitted over the channel during the last measurement period.	Number	This metric indicates the workload over a channel.
	Data received: Indicates the rate at which data is received over the channel during the last measurement period	KB/Sec	
	Data transmitted: Indicates the rate at which data is transmitted over the channel during the last measurement period	KB/Sec	

	Time since last message: Reflects the time that has elapsed since the last message was transmitted over this channel	Secs	If an unusually long time has elapsed since when the last message was transmitted, it could reflect an error in the application (e.g., a hung application that is no longer transmitting messages over the channel).

Note:

To enable the eG agent to monitor the IBM MQ server effectively, start the command server for each queue manager that you want the eG agent to monitor. A command server is mandatory for all administration involving PCFs, the MQAI, and also for remote administration. The eG agent's MQ library is a PCF. So, the command server needs to be started.

4

Monitoring FioranoMQ Servers

FioranoMQ is a communication platform that dramatically reduces the development time for network applications. It incorporates a 100% pure Java implementation of JMS, which provides a standards-based method to access distributed-system services. It further includes support for critical network services such as transactions and guaranteed message delivery.

To ensure high uptime and optimal performance of the FioranoMQ server, it is imperative that you monitor the server operations continuously.

eG Enterprise offers a specialized *Fiorano MQ* monitoring model (see Figure 4.1) , using which administrators can determine the overall health of FioranoMQ server

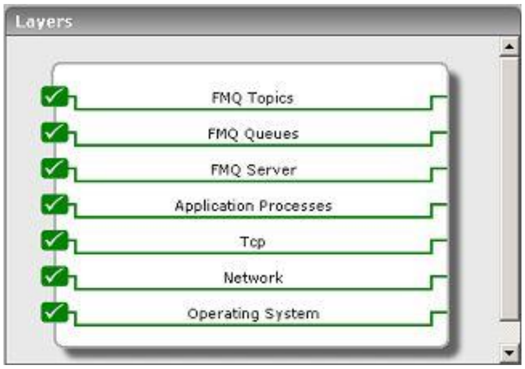


Figure 4.1: The layer model of a FioranoMQ server

The sections to come will discuss the top 3 layers only of Figure 4.1, as the remaining layers have already been discussed in detail in the *Monitoring Unix and Windows Servers* document.

Note:

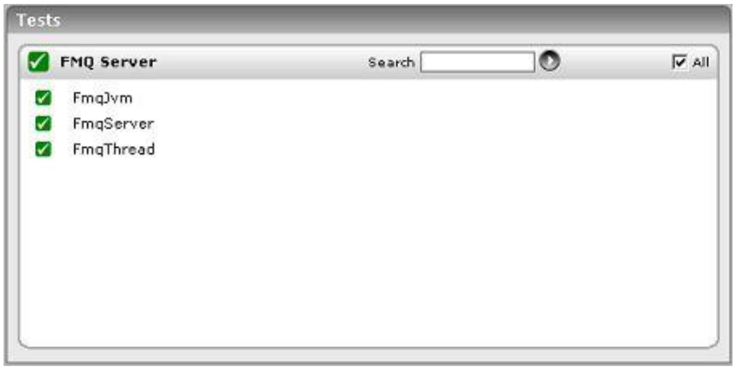
To monitor a FioranoMQ Server, events generation in the FioranoMQ Server should be enabled.

a.

4.1 The FMQ Server Layer

The tests mapped to this layer extract a wide variety of metrics from the FioranoMQ server, which reveal the following:

- Is the FioranoMQ server overloaded?
- Have adequate threads been spawned on the server to handle the load?
- Have sufficient memory resources been allocated to the server for processing requests?
- Are topics functioning smoothly?
- Do topics and queues consist of too many undelivered messages?



☑ FmqJvm
☑ FmqServer
☑ FmqThread

Figure 4.2: The tests associated with the FMQ Server layer

4.1.1 FmqThread Test

The FmqThread test reports the status of threads executed by the FioranoMQ.

Purpose	Reports the status of threads executed by the FioranoMQ
Target of the test	A FioranoMQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **HOMEDIR** – The location of the directory in which the FioranoMQ server has been installed. For example, the **HOMEDIR** for a Windows installation of the FioranoMQ server will be of the following format: `C:\PROGRA~1\Fiorano\FIORAN~1.0`. The format for a Unix installation will be: `/user/egurkha/Fiorano/FioranoMQ7.0`. 5. **SVRBINDIR** – The full path to the bin directory of the FioranoMQ server installation that contains the file `ConnectionManager.xml` in FioranoMQ server 6.0, or the `FMQListeners.xml` in the FioranoMQ 7.0. These files, which are required for starting the respective FioranoMQ servers, also help the test in determining the version number of the FioranoMQ server (whether 6 or 7). For example, the **SVRBINDIR** for a Windows installation of the server will be of the format: `C:\PROGRA~1\Fiorano\FIORAN~1.0\bin`. The format for Unix installations will be: `/user/egurkha/Fiorano/FioranoMQ7.0/bin`. 6. **SERVERMODE** - The mode in which the FioranoMQ server is running. This parameter can take one of the following values: ➢ **tcp**: In this mode, the FioranoMQ server accepts non-secure TCP connections. This is the default value for **SERVERMODE** parameter. ➢ **ssljsse**: In this mode, the FioranoMQ server accepts secure connections which are serviced using Sun's JSSE implementation. ➢ **sslphaos**: In this mode, the FioranoMQ server accepts secure TCP connections and secure connections using Phaos. ➢ **http**: In this mode, the FioranoMQ server accepts non-secure HTTP connections. ➢ **httpjsse**: In this mode, the FioranoMQ server accepts secure HTTPS connections which are serviced using Sun's JSSE implementation. ➢ **httpphaos**: In this mode, the FioranoMQ server accepts secure HTTPS connections using Phaos. 7. **ADMINID** - The user name of the FioranoMQ server's administrator. The default is "admin". 8. **ADMINPASSWORD** – The password corresponding to the specified admin user. 9. **CONFIRMPASSWORD** - Confirm the password by retyping it here. 10. **ACF** - **ACF** stands for Admin Connection Factory object. Specify the name of an existing ACF object in this text box. This object is used to obtain a handle to an Admin connection. 11. **TCF** - **TCF** stands for Topic Connection Factory object. Specify the name of an existing TCF object in this text box. This object is used to set up a connection with the provider. 12. **TRUSTSTORE** - The truststore or keystore database which the JVM uses to verify certificates. For example, this parameter can take the value `c:\FioranoMQ\bin\jssecacerts` on Windows (on Unix, the path would be expressed as: `/user/egurkha/Fiorano/FioranoMQ7.0/bin/jssecacerts`), where *jssecacerts* is the truststore database which the JVM uses.
Outputs of the test	One set of results for a FioranoMQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Active threads: The number of threads in the FioranoMQ that are currently active	Number	If the number of active threads is very high, then it indicates excessive activity on the server.
	Idle threads: The number of threads in the FioranoMQ that are currently idle	Number	

Note:

Refer to page 39 to know the values of the **SERVER MODE**, **ACF** and **TCF** test parameters.

4.1.2 FmqServer Test

The FmqServer test reports performance statistics pertaining to the FioranoMQ server.

Purpose	Reports performance statistics pertaining to the FioranoMQ server
Target of the test	A FioranoMQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **HOMEDIR** – The location of the directory in which the FioranoMQ server has been installed. For example, the **HOMEDIR** for a Windows installation of the FioranoMQ server will be of the following format: `C:\PROGRA~1\Fiorano\FIORAN~1.0`. The format for a Unix installation will be: `/user/egurkha/Fiorano/FioranoMQ7.0`. 5. **SVRBINDIR** – The full path to the bin directory of the FioranoMQ server installation that contains the file `ConnectionManager.xml` in FioranoMQ server 6.0, or the `FMQListeners.xml` in the FioranoMQ7.0. These files, which are required for starting the respective FioranoMQ servers, also help the test in determining the version number of the FioranoMQ server (whether 6 or 7). For example, the **SVRBINDIR** for a Windows installation of the server will be of the format: `C:\PROGRA~1\Fiorano\FIORAN~1.0\bin`. The format for Unix installations will be: `/user/egurkha/Fiorano/FioranoMQ7.0/bin`. 6. **SERVERMODE** - The mode in which the FioranoMQ server is running. This parameter can take any of the following values: ➢ **tcp**: In this mode, the FioranoMQ server accepts non-secure TCP connections. This is the default value for **SERVERMODE** parameter. ➢ **ssljsse**: In this mode, the FioranoMQ server accepts secure connections which are serviced using Sun's JSSE implementation. ➢ **sslphaos**: In this mode, the FioranoMQ server accepts secure TCP connections and secure connections using Phaos. ➢ **http**: In this mode, the FioranoMQ server accepts non-secure HTTP connections. ➢ **httpjsse**: In this mode, the FioranoMQ server accepts secure HTTPS connections which are serviced using Sun's JSSE implementation. ➢ **httpphaos**: In this mode, the FioranoMQ server accepts secure HTTPS connections using Phaos. 7. **ADMINID** - The user name of the FioranoMQ server's administrator. The default is "admin". 8. **ADMINPASSWORD** – The password corresponding to the specified admin user. 9. **CONFIRMPASSWORD** - Confirm the password by retyping it here. 10. **ACF** - **ACF** stands for Admin Connection Factory object. Specify the name of an existing ACF object in this text box. This object is used to obtain a handle to an Admin connection. 11. **TCF** - **TCF** stands for Topic Connection Factory object. Specify the name of an existing TCF object in this text box. This object is used to set up a connection with the provider. 12. **TRUSTSTORE** - The truststore or keystore database which the JVM uses to verify certificates. For example, this parameter can take the value `c:\FioranoMQ\bin\jssecacerts` on Windows (on Unix, the path would be expressed as: `/user/egurkha/Fiorano/FioranoMQ7.0/bin/jssecacerts`), where *jssecacerts* is the truststore database which the JVM uses.
Outputs of the test	One set of results for a FioranoMQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Active connections: The number of active client connections to the server	Number	A large number of active connections could indicate that a large number of clients have connected to the server. Alternatively, it could also mean that connections are not being released / closed properly.
	Total topics: The number of topics present in the server. The FioranoMQ server sends messages to a client through a queue or a topic. While a message in a queue can be sent to a single recipient only, a message in a topic can be sent to multiple recipients who subscribed for the topic.	Number	
	Topics messages consumed: The rate at which messages are received from all the topics on the server	Reads/Sec	This measure is an indicator of clients actively connecting and receiving messages from topics for which they have subscribed. An increase or decrease in the rate of message consumption represents the system load.
	Topic messages published: The rate of publication of messages to all the topics on the server.	Writes/Sec	This measure is an indicator of clients actively connecting and sending messages to topics. An increase or decrease in the rate of message publication represents the system load.
	Total durable subscribers: The number of durable subscriptions on all the topics present in the server. If a client with a durable subscription is not currently available for receiving messages, then such messages will not be lost in transit. Instead, the messages will be stored by the server until such time that client becomes available. Once the client is activated, the messages will be automatically sent.	Number	If the total number of durable subscribers is high, then we can expect the total durable messages to be stored on the FioranoMQ also to be relatively on the higher side.

	Total durable messages: The total number of messages pending for all durable subscribers on all topics present in the server	Number	If the value of this measure is consistently high, it indicates that the receivers are not receiving the messages stored in the topics. The reason is that the clients may not be active to receive messages or the clients may be unable to connect to the server.
	Total queues: The number of queues present in the server	Number	
	Queue messages consumed: The rate at which messages are received from all the queues on the server	Reads/Sec	This measure is an indicator of clients actively connecting and popping messages from queues. An increase or decrease in the rate of message consumption represents the system load.
	Queue messages pushed: The rate of number of messages pushed to all queues on the server.	Writes/Sec	This measure is an indicator of clients actively connecting and pushing messages into the queues. An increase or decrease in this rate represents the system load.
	Topic connection factories: The number of topic connection factories present in the server.	Number	
	Admin connection factories: The number of admin connection factories present in the server.	Number	

Note:

Refer to page 39 to know the values of the **SERVER MODE**, **ACF** and **TCF** test parameters.

4.1.3 FmqJvm Test

The FmqJvm test reports the memory usage of the FioranoMQ server.

Purpose	Reports the memory usage of the FioranoMQ server
Target of the test	A FioranoMQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **HOMEDIR** – The location of the directory in which the FioranoMQ server has been installed. For example, the **HOMEDIR** for a Windows installation of the FioranoMQ server will be of the following format: `C:\PROGRA~1\Fiorano\FIORAN~1.0`. The format for a Unix installation will be: `/user/egurkha/Fiorano/FioranoMQ7.0`. 5. **SVRBINDIR** – The full path to the bin directory of the FioranoMQ server installation that contains the file `ConnectionManager.xml` in FioranoMQ server 6.0, or the `FMQListeners.xml` in the FioranoMQ7.0. These files, which are required for starting the respective FioranoMQ servers, also help the test in determining the version number of the FioranoMQ server (whether 6 or 7). For example, the **SVRBINDIR** for a Windows installation of the server will be of the format: `C:\PROGRA~1\Fiorano\FIORAN~1.0\bin`. The format for Unix installations will be: `/user/egurkha/Fiorano/FioranoMQ7.0/bin`. 6. **SERVERMODE** - The mode in which the FioranoMQ server is running. This parameter can take any of the following values: ➢ **tcp**: In this mode, the FioranoMQ server accepts non-secure TCP connections. This is the default value for **SERVERMODE** parameter. ➢ **ssljsse**: In this mode, the FioranoMQ server accepts secure connections which are serviced using Sun's JSSE implementation. ➢ **sslphaos**: In this mode, the FioranoMQ server accepts secure TCP connections and secure connections using Phaos. ➢ **http**: In this mode, the FioranoMQ server accepts non-secure HTTP connections. ➢ **httpjsse**: In this mode, the FioranoMQ server accepts secure HTTPS connections which are serviced using Sun's JSSE implementation. ➢ **httpphaos**: In this mode, the FioranoMQ server accepts secure HTTPS connections using Phaos. 7. **ADMINID** - The user name of the FioranoMQ server's administrator. The default is "admin". 8. **ADMINPASSWORD** – The password corresponding to the specified admin user. 9. **CONFIRMPASSWORD** - Confirm the password by retyping it here. 10. **ACF** - **ACF** stands for Admin Connection Factory object. Specify the name of an existing ACF object in this text box. This object is used to obtain a handle to an Admin connection. 11. **TCF** - **TCF** stands for Topic Connection Factory object. Specify the name of an existing TCF object in this text box. This object is used to set up a connection with the provider. 12. **TRUSTSTORE** - The truststore or keystore database which the JVM uses to verify certificates. For example, this parameter can take the value `c:\FioranoMQ\bin\jssecacerts` on Windows (on Unix, the path would be expressed as: `/user/egurkha/Fiorano/FioranoMQ7.0/bin/jssecacerts`), where *jssecacerts* is the truststore database which the JVM uses.
Outputs of the test	One set of results for a FioranoMQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Memory consumed: The memory utilized by the server	MB	This measure serves as an indicator of the level of activity on the FioranoMQ server. While a high value indicates that the server is actively performing its messaging functions, a low value indicates that client connections, and the receipt and transmission of messages, is low.
	Free memory: The unused memory in the server	MB	This indicates how much more memory is available for the JVM to perform further duties. If frequently fully used up and close to 0, it means the total memory allocation to JVM has to be tuned up accordingly.
	Total memory: The total memory available to the FioranoMQ server	MB	This indicates how much memory is totally available for the JVM of FioranoMQ server.
Note: Refer to page 39 to know the values of the **SERVER MODE**, **ACF** and **TCF** test parameters.			

4.2 The FMQ Queues Layer

Use the **FmqQueue** test associated with this layer to determine how many messages pending delivery are still available in the queue.

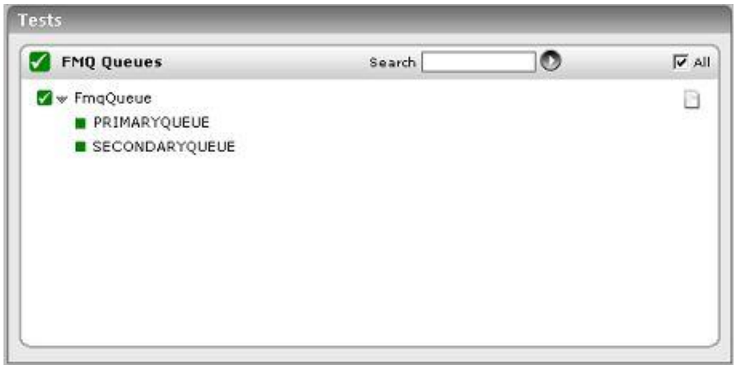


Figure 4.3: The tests associated with the FMQ Queues layer

4.2.1 FmqQueue Test

The FmqQueue test tracks various statistics pertaining to the queues on a FioranoMQ server.

Purpose	Tracks various statistics pertaining to the queues on a FioranoMQ server
Target of the test	A FioranoMQ server

<table>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The port at which the host listens
4. <b>HOMEDIR</b> – The location of the directory in which the FioranoMQ server has been installed.
5. <b>SERVERMODE</b> - The mode in which the FioranoMQ server is running. This parameter can take any of the following values:
➢ <b>tcp</b>: In this mode, the FioranoMQ server accepts non-secure TCP connections. This is the default value for <b>SERVERMODE</b> parameter.
➢ <b>ssljsse</b>: In this mode, the FioranoMQ server accepts secure connections which are serviced using Sun's JSSE implementation.
➢ <b>sslphaos</b>: In this mode, the FioranoMQ server accepts secure TCP connections and secure connections using Phaos.
➢ <b>http</b>: In this mode, the FioranoMQ server accepts non-secure HTTP connections.
➢ <b>httpjsse</b>: In this mode, the FioranoMQ server accepts secure HTTPS connections which are serviced using Sun's JSSE implementation.
➢ <b>httpphaos</b>: In this mode, the FioranoMQ server accepts secure HTTPS connections using Phaos.
6. <b>ADMINID</b> - The user name of the FioranoMQ server's administrator. The default is "admin".
7. <b>ADMINPASSWORD</b> – The password corresponding to the specified admin user.
8. <b>CONFIRMPASSWORD</b> - Confirm the password by retyping it here.
9. <b>ACF</b> - <b>ACF</b> stands for Admin Connection Factory object. Specify the name of an existing ACF object in this text box. This object is used to obtain a handle to an Admin connection.
10. <b>TCF</b> - <b>TCF</b> stands for Topic Connection Factory object. Specify the name of an existing TCF object in this text box. This object is used to set up a connection with the provider.
11. <b>TRUSTSTORE</b> - The truststore or keystore database which the JVM uses to verify certificates. For example, this parameter can take the value <i>c:|FioranoMQ|bin|jssecacerts</i>, where <i>jssecacerts</i> is the truststore database which the JVM uses.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every queue on a FioranoMQ server</td></tr>
<tr><td><b>Measurements made by the</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
</table>

test	**Total deliverable messages:** The number of messages in the queue that are awaiting delivery	Number	If the value of this measure has been consistently high, then it indicates that the receivers are not receiving the messages in the queue. This could be due to either of the following reasons: ➢ The client may not be active to receive messages ➢ The client may be unable to connect to the server
	Total undeleted messages: The number of undeleted messages in the queue. Undeleted messages are those for which the server has not received acknowledgement from the client.	Number	MQ servers mostly consider a message as delivered only if an acknowledgement is received from the client after delivery. Undeleted messages are those messages for which the MQ server has not received the acknowledgements from the client, and are waiting for deletion from the queue. Too many undeleted messages could unnecessarily burden the server and seriously hamper performance. Alternately it can mean that due to some reasons the clients are not able to connect/send acknowledgement to the FioranoMQ server.

Note:

Refer to page 39 to know the values of the **SERVER MODE**, **ACF** and **TCF** test parameters.

4.3 The FMQ Topics Layer

This layer helps administrators quickly determine processing delays in the topics (see Figure 4.4).

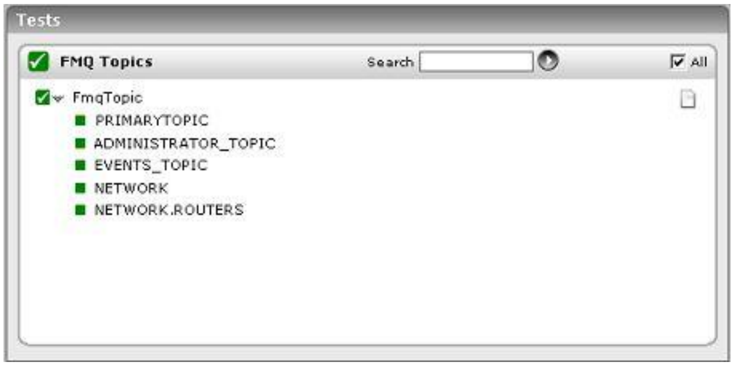


Figure 4.4: The tests associated with the FMQ Topics layer

4.3.1 FmqTopic Test

The FmqTopic test reports general statistics pertaining to the topics on a FioranoMQ server.

Purpose	Reports general statistics pertaining to the topics on a FioranoMQ server
Target of the test	A FioranoMQ server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The port at which the host listens 4. **HOMEDIR** – The location of the directory in which the FioranoMQ server has been installed. For example, the **HOMEDIR** for a Windows installation of the FioranoMQ server will be of the following format: `C:\PROGRA~1\Fiorano\FIORAN~1.0`. The format for a Unix installation will be: `/user/egurkha/Fiorano/FioranoMQ7.0`. 5. **SVRBINDIR** – The full path to the bin directory of the FioranoMQ server installation that contains the file `ConnectionManager.xml` in FioranoMQ server 6.0, or the `FMQListeners.xml` in the FioranoMQ7.0. These files, which are required for starting the respective FioranoMQ servers, also help the test in determining the version number of the FioranoMQ server (whether 6 or 7). For example, the **SVRBINDIR** for a Windows installation of the server will be of the format: `C:\PROGRA~1\Fiorano\FIORAN~1.0\bin`. The format for Unix installations will be: `/user/egurkha/Fiorano/FioranoMQ7.0/bin`. 6. **SERVERMODE** - The mode in which the FioranoMQ server is running. This parameter can take any of the following values: ➢ **tcp**: In this mode, the FioranoMQ server accepts non-secure TCP connections. This is the default value for **SERVERMODE** parameter. ➢ **ssljsse**: In this mode, the FioranoMQ server accepts secure connections which are serviced using Sun's JSSE implementation. ➢ **sslphaos**: In this mode, the FioranoMQ server accepts secure TCP connections and secure connections using Phaos. ➢ **http**: In this mode, the FioranoMQ server accepts non-secure HTTP connections. ➢ **httpjsse**: In this mode, the FioranoMQ server accepts secure HTTPS connections which are serviced using Sun's JSSE implementation. ➢ **httpphaos**: In this mode, the FioranoMQ server accepts secure HTTPS connections using Phaos. 7. **ADMINID** - The user name of the FioranoMQ server's administrator. The default is "admin". 8. **ADMINPASSWORD** – The password corresponding to the specified admin user. 9. **CONFIRMPASSWORD** - Confirm the password by retyping it here. 10. **ACF** - **ACF** stands for Admin Connection Factory object. Specify the name of an existing ACF object in this text box. This object is used to obtain a handle to an Admin connection. 11. **TCF** - **TCF** stands for Topic Connection Factory object. Specify the name of an existing TCF object in this text box. This object is used to set up a connection with the provider. 12. **TRUSTSTORE** - The truststore or keystore database which the JVM uses to verify certificates. For example, this parameter can take the value `c:\FioranoMQ\bin\jssecacerts` on Windows (on Unix, the path would be expressed as: `/user/egurkha/Fiorano/FioranoMQ7.0/bin/jssecacerts`), where *jssecacerts* is the truststore database which the JVM uses.
Outputs of the test	One set of results for every topic on a FioranoMQ server

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Total durable subscribers: The total durable subscribers for this topic. Messages meant for a durable subscriber are stored in the persistent cache even when the subscriber is inactive. These messages are delivered for durable subscribers when they connect to the FioranoMQ server.	Number	If the total number of durable subscribers is high, then we can expect the total durable messages to be stored on the FioranoMQ also to be relatively on the higher side.
	Total deliverable messages: The number of messages deliverable (undelivered+redeliverable) to the durable subscribers of this topic	Number	If the value of this measure is consistently high, it indicates that the receivers are not receiving the messages stored in the topic. The reason is that the client may not be active to receive messages or the client may be unable to connect to the server. The detailed diagnosis of this measure, if enabled, provides the details of the messages still to be delivered to a particular topic. The detailed diagnosis of this measure, if enabled, reveals information pertaining to the messages still to be delivered to a particular topic. The details displayed include the subscriber to which the message(s) is addressed to, the client from which the message(s) originated, and the number of undelivered messages.
Note: Refer to page 39 to know the values of the **SERVER MODE**, **ACF** and **TCF** test parameters.			

To know the values for **SERVERMODE**, **ACF** and **TCF** parameters for a FioranoMQ server 6.0, first, open the **ConnectionManager.xml** file in the **<FIORANO_INSTALL_DIR>/bin** directory. You will find the following section within:

```
<FMQConnectionFactories>

   <ClientConnectionFactories>

         <ConnectionFactoryInfo type="HTTPS_SUN">

               <URL>http://localhost:1856</URL>

               <SSLVendor>Sun</SSLVendor>

<SecurityManager>fiorano.jms.ex.sm.def.DefaultJSSESecurityManager</SecurityManager>

<ConnectionManager>
```

```
<ClassName>fiorano.jms.cm.def.SocketReadHandlerDefImpl</ClassName>

                        </ConnectionManager>

<ConnectionFactoryName fmq_type='TOPIC'>primaryTCF</ConnectionFactoryName>

                        <ConnectionFactoryName
fmq_type='TOPIC'>secondaryTCF</ConnectionFactoryName>

                        <ConnectionFactoryName
fmq_type='QUEUE'>primaryQCF</ConnectionFactoryName>

                        <ConnectionFactoryName
fmq_type='QUEUE'>secondaryQCF</ConnectionFactoryName>

                 <ConnectionFactoryName
fmq_type='UNIFIED'>primaryCF</ConnectionFactoryName>
</ConnectionFactoryInfo>

        </ClientConnectionFactories>

   <AdminConnectionFactories>

          <ConnectionFactoryInfo type="HTTPS_SUN">

                    <URL>http://localhost:1857</URL>

             <SSLVendor>Sun</SSLVendor>

<SecurityManager>fiorano.jms.ex.sm.def.DefaultJSSESecurityManager</SecurityManager>

                   <ConnectionManager>

                       <ClassName>
fiorano.jms.cm.def.SocketReadHandlerDefImpl</ClassName>

                   </ConnectionManager>

<ConnectionFactoryName fmq_type='ADMIN'>primaryACF</ConnectionFactoryName>

          </ConnectionFactoryInfo>

   </AdminConnectionFactories>

</FMQConnectionFactories>
```

Note the lines marked in red. The first of these lines reads as follows: **`<ConnectionFactoryInfo type="HTTPS_SUN">`**. If the **`ConnectionFactoryInfo type`** is **`HTTPS_SUN`**, then the **SERVERMODE** will be **`httpjsse`**. For the other **SERVERMODES**, the corresponding **`ConnectionFactoryInfo type`** will be:

CONNECTIONFACTORYINFO TYPE	SERVERMODE
HTTP	http
HTTPS_PHAOS	Httpphaos
SUN_SSL	Ssljsse
PHAOS_SSL	Sslphaos

The next line marked in red, reads as follows: **`<ConnectionFactoryName fmq_type='TOPIC'>primaryTCF</ConnectionFactoryName>`**. Here, the term **`primaryTCF`** indicates the name of the **TCF** object, and the same has to be provided as the **TCF** parameter.

The last line marked in red, reads as follows: **`<ConnectionFactoryName fmq_type='ADMIN'>primaryACF</ConnectionFactoryName>`**. Here, the term **`primaryACF`** indicates the name of the **ACF** object, and the same has to be provided as the **ACF** parameter.

To know the values for **SERVERMODE**, **ACF** and **TCF** for a FioranoMQ server 7.0, open the **`AdminTool.xml`** file in the **`<FIORANO_INSTALL_DIR>/bin`** directory. You will find the following section within:

```
<AdminToolCfg>
    <Param Name="LOG_MANAGER">fiorano.jms.log.def.LogManagerImpl</Param>
         <Param Name="REPEATER_ENABLED">yes</Param>
    <Param Name="REPEATER_TOPIC">REPEATER_QUERY_TOPIC</Param>
    <Param Name="REPEATER_REQUEST_TIMEOUT">10000</Param>
    <Param Name="ConfigurationTab">true</Param>
    <Param Name="showalltracecomponents">true</Param>
    <Param Name="TransportProtocol">HTTP</Param>
    <Param Name="SYSTEM_MESSAGESNOOPER_TOPIC">SYSTEM_MESSAGESNOOPER_TOPIC</Param>

    <Param Name="SYSTEM_MESSAGESNOOPER_QUEUE">SYSTEM_MESSAGESNOOPER_QUEUE</Param>
    <FioranoMQServer  URL="http://localhost:1856">
           <AdminCF Name="primaryACF"></AdminCF>
           <TopicCF Name="primarytcf"></TopicCF>
           <QueueCF Name="primaryqcf"></QueueCF>
    </FioranoMQServer>
</AdminToolCfg>
```

Note the lines marked in red. The first of these lines reads as follows: **`<Param Name="TransportProtocol">HTTP</Param>`**. If the **`TransportProtocol`** is **`HTTP`** then the **SERVERMODE** will be **`http`**. For the other **SERVERMODES**, the corresponding **`TransportProtocol`** will be:

TRANSPORTPROTOCOL	SERVERMODE
HTTPS_SUN	httpjsse
HTTPS_PHAOS	httpphaos
SUN_SSL	ssljsse
PHAOS_SSL	sslphaos

The next line marked in red, reads as follows: **`<AdminCF Name="primaryACF"></AdminCF>`**. Here, the term **`primaryACF`** indicates the name of the **ACF** object, and the same has to be provided as the **TCF** parameter.

The last line marked in red, reads as follows: **`<TopicCF Name =”primaryTCF”></TopicCF>`**. Here, the term **`primaryTCF`** indicates the name of the **TCF** object, and the same has to be provided as the **TCF** parameter.

5

Monitoring iPlanet/SunONE Messaging Servers

The Sun ONE Application suite offers a comprehensive list of products for Internet infrastructures, i.e., web server, middleware application server, LDAP server, messaging server, and identity server, that are used in many domains such as banking, trading, healthcare, and logistics to support mission-critical services. IT infrastructures based on the Sun ONE Application suite follow the popular multi-tier architecture wherein the web server functions as the front-end receiving client requests, the application server hosts the business logic components, the identity server manages user policies, the directory server handles access rights and other user information lookups, and the database server stores and retrieves application data.

Routine monitoring of the infrastructure including the network, system, and application is imperative to ensure that the infrastructure functions at peak performance at all times. Since each Sun ONE application performs a different, specialized function, the monitoring has to be specific to each application – e.g., is the mail server delivering emails? is the application server's heap effectively sized?. More importantly, since the different Sun ONE applications inter-operate to support the end-user service, it is critical that the monitoring system track the inter-dependencies between applications in order to pin-point the exact source of a performance bottleneck in the infrastructure.

The eG Sun ONE monitor offers extensive infrastructure monitoring capabilities for the Sun ONE application suite. Pre-built models for Sun ONE web, application, directory, and messaging servers metrics, and how the metrics are to be correlated in order to assist with problem diagnosis.

Using the customized model (see Figure 5.1) that eG Enterprise presents for the SunONE messaging server, the following can be tracked:

- External measure of SMTP availability to accept messages, measure of POP3/IMAP availability, number of outstanding e-mails, and round trip time for mail delivery
- Monitoring of user activity, failed logins, internal message queue lengths, request processing and failure rates, messaging store database activity monitoring including transactions, deadlocks, etc.

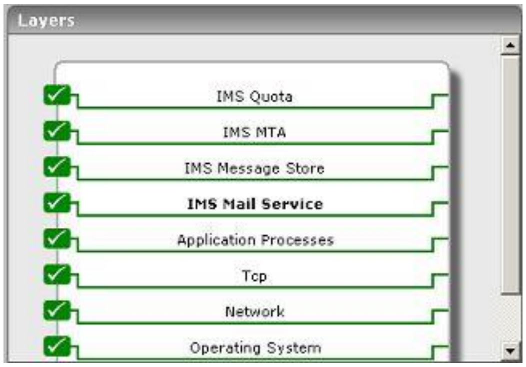


Figure 5.1: Layer model of the iPlanet/SunONE messaging server

The sub-sections to come will discuss only the top 4 layers depicted by Figure 5.1. The remaining layers have already been dealt with in the *Monitoring Unix and Windows Servers* document.

5.1 The IMS Mail Service Layer

The tests associated with this layer enable a fair assessment of the efficiency of the HTTP, IMAP, and POP3 services offered by the messaging server, and the availability and responsiveness of the IMAP, LDAP, and POP3 ports.

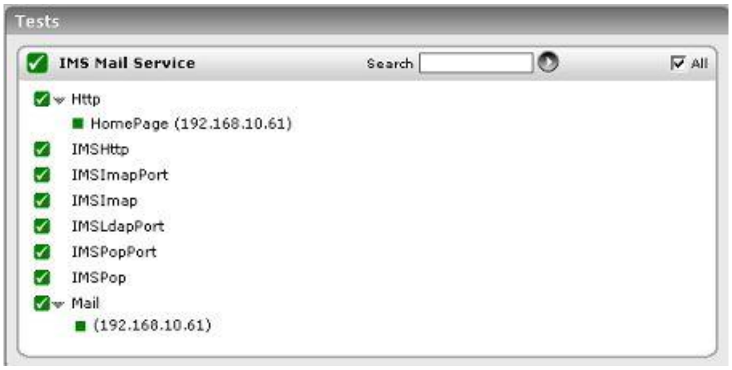


Figure 5.2: The tests associated with the IMS Mail Service layer

The Http test and Mail test depicted by Figure 5.2 have been discussed elaborately in the *Monitoring Web Servers* and the *Monitoring Mail Servers* documents, respectively. However, the difference lies in how a few parameters of these tests are configured for the iPlanet/SunONE messaging servers.

The Http test requires that a **URL** parameter be configured. In the **URL** text box, the web page(s) that is being accessed has to be provided. While multiple URLs (separated by commas) can be provided, each URL should be of the format **URL name:URL value**. **URL name** is a unique name assigned to the URL, and the **URL value** is the value of the URL. By default, the URL text box will display **HomePage:http://iplanetmessagingserverIPorhostname:defaultportoftheHttpserver/**. For example, if the IP of the iPlanet messaging server is 192.168.10.47, then the URL displayed will be - HomePage:http://192.168.10.47:80/, where "80" is the default port of an HTTP server. If the Http port of the iPlanet messaging server is not "80", then you need to change the port in this display, accordingly. In other words, if "81" is the Http port of the iPlanet messaging server, then you need to change the URL to **HomePage:http://192.168.10.47:81/**. Similarly, the **PORT** parameter should be configured with the SMTP port of messaging server.

Likewise, the **SENDPORT** parameter of the Mail test should be configured with the SMTP port of the messaging server.

5.1.1 IMSHttp Test

The IMSHttp test monitors the HTTP service supported by the iPlanet messaging server.

<table>
<tr><td><b>Purpose</b></td><td colspan="3">Monitors the HTTP service supported by the iPlanet messaging server</td></tr>
<tr><td><b>Target of the test</b></td><td colspan="3">An iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>SERVERROOT</b> – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: <i>C:|iplanet|server5</i>. In Unix platforms, the path can be specified in the following format: <i>/usr/iplanet/server5</i>. A server group may include other iPlanet servers in addition to the messaging server.
5. <b>VERSION</b> - This refers to the version number of the SunONE messaging server that is being monitored. By default, <i>none</i> will be displayed as the <b>VERSION</b>. If you are monitoring a SunONE messaging server that is of a version below 6.0, you need not change the default value of this parameter. However, while monitoring version 6.0 or above, the exact version number needs to be explicitly mentioned against this parameter.
6. <b>COUNTERREGISTRY</b> - Enter the full path to the counter registry to use. By default, the path to the counter registry will be: <b><code><IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE>/counter/counter</code></b>. Here, <b><code>SERVER_ROOT_DIR</code></b> will be the value of the <b>SERVERROOT</b> parameter above, and the <b><code>SERVER_INSTANCE</code></b> is the name of the instance of the iPlanet messaging server specified during installation. For example, in Windows environments, the path specification can be: <i>C:\iPlanet\Server5\msg-egtes\counter\counter</i>, and in Unix environments, it can be: <i>usr/iplanet/server5/msg-sun08/counter/counter</i>.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td rowspan="3"><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td><b>Active connections:</b>
Indicates the number of active HTTP connections to the server.</td><td>Number</td><td>A high value for this measure indicates that a large number of HTTP connections are opened to the server.</td></tr>
<tr><td><b>Active sessions:</b>
Indicates the number of HTTP server sessions(login to logout) currently active.</td><td>Number</td><td>A high value for this measure indicates that a large number of HTTP clients are logged into the server.</td></tr>
</table>

	Connections served: Indicates the rate of connections served by the HTTP server	Conns/Sec	A high value for this measure indicates that the messaging server is in good health.
	Good logins: Indicates the rate of successful logins served by the HTTP server	Logins/Sec	
	Failed logins: Indicates the rate of failed logins served by the HTTP server	Logins/Sec	
	Failed connections: Indicates the rate at which Http connections failed	Conns/Sec	

5.1.2 IMSImap Test

The IMSImap test reports critical metrics pertaining to the performance of the IMAP service of the iPlanet Messaging server.

Purpose	Reports critical metrics pertaining to the performance of the IMAP service of the iPlanet Messaging server
Target of the test	An iPlanet/SunONE messaging server
Agent deploying the test	An internal agent

<table>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>SERVERROOT</b> – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: <i>C:|iplanet|server5</i>. In Unix platforms, the path can be specified in the following format: <i>/usr/iplanet/server5</i>. A server group may include other iPlanet servers in addition to the messaging server.
5. <b>VERSION</b> - This refers to the version number of the SunONE messaging server that is being monitored. By default, <i>none</i> will be displayed as the <b>VERSION</b>. If you are monitoring a SunONE messaging server that is of a version below 6.0, you need not change the default <i>none</i> value of this parameter. However, while monitoring version 6.0 or above, the exact version number needs to be explicitly mentioned against this parameter.
6. <b>COUNTERREGISTRY</b> - Enter the full path to the counter registry to use. By default, the path to the counter registry will be: <b><code><IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE>/counter/counter</code></b>. Here, <b><code>SERVER_ROOT_DIR</code></b> will be the value of the <b>SERVERROOT</b> parameter above, and the <b><code>SERVER_INSTANCE</code></b> is the name of the instance of the iPlanet messaging server specified during installation. For example, in Windows environments, the path specification can be: <i>C:\iPlanet\Server5\msg-egtes\counter\counter</i>, and in Unix environments, it can be: <i>usr/iplanet/server5/msg-sun08/counter/counter</i>.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td></td><td><b>Active connections:</b>
Indicates the number of active IMAP connections to the server.</td><td>Number</td><td>A high value for this measure indicates that a large number of IMAP connections are opened to the server.</td></tr>
<tr><td></td><td><b>Connections served:</b>
Indicates the rate of connections served by the IMAP server</td><td>Conns/Sec</td><td>A high value for this measure indicates that the IMAP service is functioning efficiently.</td></tr>
<tr><td></td><td><b>Good logins:</b>
Indicates the rate of successful logins served by the IMAP server</td><td>Logins/Sec</td><td></td></tr>
<tr><td></td><td><b>Failed logins:</b>
Indicates the rate of failed logins served by the IMAP server</td><td>Logins/Sec</td><td></td></tr>
<tr><td></td><td><b>Failed connections:</b>
Indicates the rate at which IMAP connections failed</td><td>Conns/Sec</td><td></td></tr>
</table>

5.1.3 IMSPop Test

The IMSPop test reports performance statistics related to the POP3 service of the iPlanet Messaging server.

<table>
<tr><td><b>Purpose</b></td><td colspan="3">Reports performance statistics related to the POP3 service of the iPlanet Messaging server</td></tr>
<tr><td><b>Target of the test</b></td><td colspan="3">An iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>SERVERROOT</b> – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: <i>C:|iplanet|server5</i>. In Unix platforms, the path can be specified in the following format: <i>/usr/iplanet/server5</i>. A server group may include other iPlanet servers in addition to the messaging server.
5. <b>VERSION</b> - This refers to the version number of the SunONE messaging server that is being monitored. By default, <i>none</i> will be displayed as the <b>VERSION</b>. If you are monitoring a SunONE messaging server that is of a version below 6.0, you need not change the default <i>none</i> value of this parameter. However, while monitoring version 6.0 or above, the exact version number needs to be explicitly mentioned against this parameter.
6. <b>COUNTERREGISTRY</b> - Enter the full path to the counter registry to use. By default, the path to the counter registry will be: <b><code><IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE>/counter/counter</code></b>. Here, <b><code>SERVER_ROOT_DIR</code></b> will be the value of the <b>SERVERROOT</b> parameter above, and the <b><code>SERVER_INSTANCE</code></b> is the name of the instance of the iPlanet messaging server specified during installation. For example, in Windows environments, the path specification can be: <i>C:\iPlanet\Server5\msg-egtes\counter\counter</i>, and in Unix environments, it can be: <i>usr/iplanet/server5/msg-sun08/counter/counter</i>.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td></td><td><b>Active connections:</b>
Indicates the number of active POP3 connections to the server.</td><td>Number</td><td>A high value for this measure indicates that a large number of POP3 connections are opened to the server.</td></tr>
<tr><td></td><td><b>Connections served:</b>
Indicates the rate of connections served by the POP3 server</td><td>Conns/Sec</td><td>A high value for this measure indicates that the POP3 service is functioning efficiently.</td></tr>
</table>

	Good logins: Indicates the rate of successful logins served by the POP3 server	Logins/Sec	
	Failed logins: Indicates the rate of failed logins served by the POP3 server	Logins/Sec	
	Failed connections: Indicates the rate at which POP3 connections failed	Conns/Sec	

5.1.4 IMSLdapPort Test

This test monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's LDAP service.

Purpose	Monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's LDAP service		
Target of the test	An iPlanet/SunONE messaging server		
Agent deploying the test	An external agent		
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server 4. **TARGETPORTS** –Specify the port at which the LDAP service of the iPlanet messaging server listens.		
Outputs of the test	One set of results for every iPlanet/SunONE messaging server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	LDAP availability: Indicates whether the TCP connection is available or not.	Percent	An availability problem can be caused by different factors - e.g., the server process may not be up, or, a network problem may exist, etc.
	LDAP response time: Indicates the time taken (in seconds) by the server to respond to a request.	Secs	An increase in response time can be caused by several factors such as a server bottleneck, a network problem, etc.

5.1.5 IMSImapPort Test

This test monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's IMAP service.

Purpose	Monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's IMAP service		
Target of the test	An iPlanet/SunONE messaging server		
Agent deploying the test	An external agent		
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server 4. **TARGETPORTS** –Specify the port at which the IMAP service of the iPlanet messaging server listens.		
Outputs of the test	One set of results for every iPlanet/SunONE messaging server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	IMAP availability: Indicates whether the TCP connection is available or not.	Percent	An availability problem can be caused by different factors - e.g., the server process may not be up, or, a network problem may exist, etc.
	IMAP response time: Indicates the time taken (in seconds) by the server to respond to a request.	Secs	An increase in response time can be caused by several factors such as a server bottleneck, a network problem, etc.

5.1.6 IMSPopPort Test

This test monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's IMAP service.

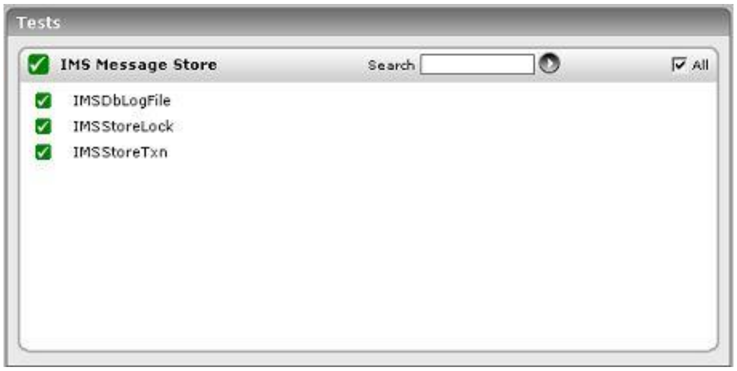
Purpose	Monitors the availability and responsiveness of the TCP port of the iPlanet messaging server's POP3 service
Target of the test	An iPlanet/SunONE messaging server
Agent deploying the test	An external agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server 4. **TARGETPORTS** –Specify the port at which the POP3 service of the iPlanet messaging server listens.		
Outputs of the test	One set of results for every iPlanet/SunONE messaging server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	POP availability: Indicates whether the TCP connection is available or not.	Percent	An availability problem can be caused by different factors - e.g., the server process may not be up, or, a network problem may exist, etc.
	POP response time: Indicates the time taken (in seconds) by the server to respond to a request.	Secs	An increase in response time can be caused by several factors such as a server bottleneck, a network problem, etc.

b.

5.2 The IMS Message Store Layer

The tests associated with this proactively alerts administrators of log file accumulation that can choke the database, abnormal locking behaviour that can result in deadlocks, and a high number of transaction aborts indicative of a performance bottleneck on the messaging server.



☑ IMS Message Store

☑ IMSDbLogFile

☑ IMSStoreLock

☑ IMSStoreTxn

Figure 5.3: The tests associated with the IMS Message Store layer

5.2.1 IMSDbLogFile Test

The iPlanet messaging server instance message store contains a database (Berkley DB) that stores information about the mailboxes on the server, and stores quota information about the mailboxes. The IMSDbLogFileTest monitors the log files of the message store database.

Purpose	Monitors the log files of the message store database

<table>
<tr><td><b>Target of the test</b></td><td colspan="3">An iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>MBOXLISTPATH</b> – The complete path to the "mboxlist" directory of the current instance of the messaging server. By default, this directory will be located within the "store" directory of the "messaging server instance directory". For example, the <b>MBOXLISTPATH</b> in Windows environments can be: <i>C:\iPlanet\Server5\msg-egtest\store\mboxlist</i>, and in Unix environments, it can be: <i>/usr/iplanet/server5/msg-sun08/store/mboxlist.</i></td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td></td><td><b>Number of files:</b>
Indicates the number of database log files present in the directory, <b><IPLANET_MESSAGING_SERVER_INSTALL_DIR>/<CURRENT_INSTANCE_DIRECTORY>/store/mboxlist</b>. Example of the log file is <i>log.0000000014</i>.</td><td>Number</td><td>Database log files refer to sleepycat transaction checkpointing log files (in the directory, <b><IPLANET_MESSAGING_SERVER_INSTALL_DIR>/<CURRENT_INSTANCE_DIRECTORY>/store/mboxlist</b>). If log files accumulate, then database checkpointing will not occur. At any given point of time, the presence of 2 or 3 log files is indicative of good health. If there are more files, it could be a sign of a problem. Log file accumulation could also be due to "stored" problems.</td></tr>
<tr><td></td><td><b>Total size of files:</b>
Indicates the space occupied by the database log files.</td><td>MB</td><td></td></tr>
</table>

5.2.2 IMSStoreLock Test

The iPlanet messaging server instance message store contains a database (Berkley DB) that stores information about the mailboxes on the server and stores quota information about the mailboxes. The state of DB-locks is held by different server processes. These database locks can affect the performance of the message store. The IMSStoreLockTest monitors the behaviour of the database locks.

Purpose	Monitors the behaviour of the database locks
Target of the test	An iPlanet/SunONE messaging server

<table>
<tr><td>**Agent deploying the test**</td><td colspan="3">An internal agent</td></tr>
<tr><td>**Configurable parameters for the test**</td><td colspan="3">1. **TEST PERIOD** – How often should the test be executed
2. **HOST** - The host for which the test is being configured
3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
4. **SERVERROOT** – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: *C:|iplanet|server5*. In Unix platforms, the path can be specified in the following format: */usr/iplanet/server5*. A server group may include other iPlanet servers in addition to the messaging server.
5. **COUNTERREGISTRY** - Enter the full path to the counter registry to use. By default, the path to the counter registry will be: **`<IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE>/counter/counter`**. Here, **`SERVER_ROOT_DIR`** will be the value of the **SERVERROOT** parameter above, and the **`SERVER_INSTANCE`** is the name of the instance of the iPlanet messaging server specified during installation. For example, in Windows environments, the path specification can be: *C:|iPlanet|Server5|msg-egtes|counter|counter*, and in Unix environments, it can be: *usr/iplanet/server5/msg-sun08/counter/counter*.</td></tr>
<tr><td>**Outputs of the test**</td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td rowspan="3">**Measurements made by the test**</td><td>**Measurement**</td><td>**Measurement Unit**</td><td>**Interpretation**</td></tr>
<tr><td>**Lock requests:**
Indicates the rate of database lock requests.</td><td>Reqs/Sec</td><td></td></tr>
<tr><td>**Lock releases:**
Indicates the rate of database lock releases.</td><td>Reqs/Sec</td><td></td></tr>
<tr><td></td><td>**Deadlocks occurred:**
The rate of deadlocks</td><td>Deadlocks/Sec</td><td>A non zero value for this measure is indicative of the occurrence of a deadlock in the message store database. In case of deadlocks, messages will not be getting inserted into the store at reasonable speeds and the ims-ms channel queue will grow larger as a result. The stored utility will automatically perform Deadlock detection and rollback of deadlocked database transactions.</td></tr>
</table>

5.2.3 IMSStoreTxn Test

The IMSStoreTxn test monitors the transactions to the message store database.

Purpose	Monitors the transactions to the message store database
Target of the	An iPlanet/SunONE messaging server

<table>
<tr><td><b>test</b></td><td colspan="3"></td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>SERVERROOT</b> – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: <i>C:|iplanet|server5</i>. In Unix platforms, the path can be specified in the following format: <i>/usr/iplanet/server5</i>. A server group may include other iPlanet servers in addition to the messaging server.
5. <b>COUNTERREGISTRY</b> - Enter the full path to the counter registry to use. By default, the path to the counter registry will be: <b><IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE>/counter/counter</b>. Here, <b>SERVER_ROOT_DIR</b> will be the value of the <b>SERVERROOT</b> parameter above, and the <b>SERVER_INSTANCE</b> is the name of the instance of the iPlanet messaging server specified during installation. For example, in Windows environments, the path specification can be: <i>C:\iPlanet\Server5\msg-egtes\counter\counter</i>, and in Unix environments, it can be: <i>usr/iplanet/server5/msg-sun08/counter/counter</i>.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every iPlanet/SunONE messaging server</td></tr>
<tr><td><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td></td><td><b>Active transactions:</b>
Indicates the number of transactions that are currently active.</td><td>Number</td><td>A high value for this measure may indicate a large number of active transactions.</td></tr>
<tr><td></td><td><b>Transaction begins:</b>
Indicates the rate at which transactions have begun.</td><td>Trans/Sec</td><td></td></tr>
<tr><td></td><td><b>Transaction commits:</b>
Indicates the rate at which transactions have been committed</td><td>Trans/Sec</td><td></td></tr>
<tr><td></td><td><b>Transaction aborts:</b>
Indicates the rate at which transactions have been aborted/rolled back</td><td>Trans/Sec</td><td>Rollbacks are expensive operations on a database, and hence, will have to be kept at a minimum.</td></tr>
</table>

5.3 The IMS MTA Layer

Using the tests associated with this layer, you can continuously monitor message traffic to identify undelivered messages and message queues that grow consistently.

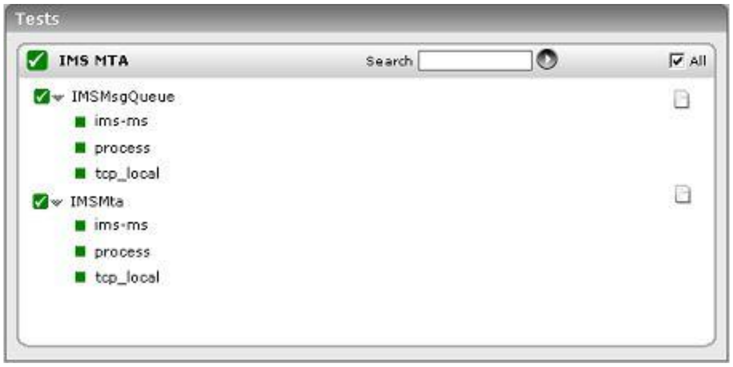


Figure 5.4: The tests associated with the IMS MTA layer

5.3.1 IMSMsgQueue Test

The channel is the fundamental MTA component that processes a message in an iPlanet messaging server. A channel represents a connection with another computer system or group of systems. The actual hardware connection or software transport or both may vary widely from one channel to the next. Messages are enqueued by channels on the way into the MTA and dequeued on the way out. A channel might dequeue a message, process the message, or enqueue the message to another MTA channel.

This test will track the queued messages and .HELD messages in a channel's queue directories.

Purpose	Tracks the queued messages and .HELD messages in the channel queue directories		
Target of the test	An iPlanet/SunONE messaging server		
Agent deploying the test	An internal agent		
Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server 4. **VERSION** - This refers to the version number of the SunONE messaging server that is being monitored. By default, *none* will be displayed as the **VERSION**. If you are monitoring a SunONE messaging server that is of a version below 6.0, you need not change the default value of this parameter. However, while monitoring version 6.0 or above, the exact version number needs to be explicitly mentioned against this parameter. 5. **INSTANCEDIRECTORY** –If you are monitoring a SunONE messaging server that is of a version below 6.0, then specify the full path to the directory that corresponds to the current messaging server instance. For example, in Windows environments, the path can be expressed as: *C:|iPlanet|Server5|msg-egtest*. In Unix platforms, the path can be specified in the following format: */usr/iplanet/server5/msg-sun08*. On the other hand, if you are monitoring version 6.0 or above of a SunONE messaging server, then ensure that the SunONE messaging server's root directory is specified as the **INSTANCEDIRECTORY**.		
Outputs of the test	One set of results for every channel in the iPlanet/SunONE messaging server		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Messages attempted:** Indicates the number of queued message files in a particular channel's queue directories.	Number	A consistent high value in this measure is an indication that messages are not being delivered, are being delayed in their delivery, or are coming in faster than the system can deliver them. This may be caused by a number of reasons such as a denial of service attack caused by a huge number of messages flooding your system, or the Job Controller not running. Common symptoms of message queue problems are: ➢ Disk space usage grows ➢ User not receiving messages in a reasonable time ➢ Message queue sizes are abnormally high
	Messages held: Indicates the number of .HELD message files in a specific channel's queue directories.	Number	If the MTA detects that messages are bouncing between servers or channels, then delivery is halted and the messages are stored in a file with the suffix .HELD in **`<IPLANET_MESSAGING_SERVER_ROOT_DIR>/<SERVER_INSTANCE_DIR>/imta/queue/channel`**. Whenever you find a value for this measure you have to diagnose and clean up .HELD messages.
	Space consumed: Indicates the total space occupied by the queued and .HELD message files	MB	

5.3.2 IMSMta Test

This test will report statistics related to the message traffic in channels.

Purpose	Tracks the queued messages and .HELD messages in the channel queue directories
Target of the test	An iPlanet/SunONE messaging server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed 2. **HOST** - The host for which the test is being configured 3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server 4. **VERSION** - This refers to the version number of the SunONE messaging server that is being monitored. By default, *none* will be displayed as the **VERSION**. If you are monitoring a SunONE messaging server that is of a version below 6.0, you need not change the default value of this parameter. However, while monitoring version 6.0 or above, the exact version number needs to be explicitly mentioned against this parameter. 5. **INSTANCEDIRECTORY** –If you are monitoring a SunONE messaging server that is of a version below 6.0, then specify the full path to the directory that corresponds to the current messaging server instance. For example, in Windows environments, the path can be expressed as: *C:\iPlanet\Server5\msg-egtest*. In Unix platforms, the path can be specified in the following format: */usr/iplanet/server5/msg-sun08*. On the other hand, if you are monitoring version 6.0 or above of a SunONE messaging server, then ensure that the SunONE messaging server's root directory is specified as the **INSTANCEDIRECTORY**.		
Outputs of the test	One set of results for every channel in the iPlanet/SunONE messaging server		
Measurements made by the test	**Measurement**	**Measurement Unit**	**Interpretation**
	Messages delivered: Indicates the number of messages which have been processed (dequeued) by the channel since the last measurement. A dequeue operation may either correspond to a successful delivery, or to a dequeue due to the message being returned to the sender.	Number	A high value of this measure indicates that the server is healthy.
	Messages attempted: Indicates the number of messages which have experienced temporary problems in dequeuing from message queue since the last measurement.	Number	
	Messages rejected: Indicates the number of attempted enqueues which have been rejected by the slave channel program since the last measurement.	Number	

	Messages failed: Indicates the number of attempted dequeues which have failed since the last measurement.	Number	This measure refers to recipient addresses that were rejected on attempted dequeue (rejection by Master channel program), or generation of a failure/bounce message.

5.4 The IMS Quota Layer

To monitor users to the server domain and their resource usage on the server, use the IMSUser test associated with this layer (see Figure 5.5).

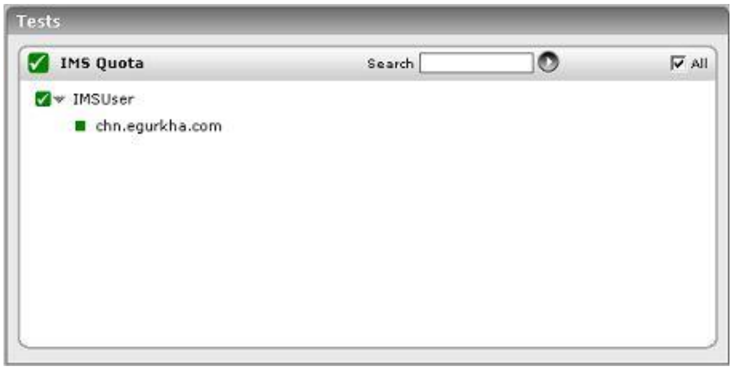


Figure 5.5: The tests associated with the IMS Quota layer

5.4.1 IMSUser Test

This test monitors the user accounts that exist in a domain.

Purpose	Monitors the accounts that exist in a domain
Target of the test	An iPlanet/SunONE messaging server
Agent deploying the test	An internal agent

<table>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>SERVERROOT</b> – Specify the path to the directory into which all servers of a given server group (i.e., all servers managed by a given Administration Server) are installed. For example, in Windows environments, the path can be expressed as: <i>C:|iplanet|server5</i>. In Unix platforms, the path can be specified in the following format: <i>/usr/iplanet/server5</i>. A server group may include other iPlanet servers in addition to the messaging server.
5. <b>CONFIGROOT</b> - Specify the path to the directory in which the config root file "msg.conf" exists. By default, this file will be located within the "config" directory of the "current messaging server instance directory". For example, in Windows environments, the path can be expressed as: <i>C:|iPlanet|Server5|msg-egtest|config</i>. In Unix platforms, the path can be specified in the following format: <i>usr/iplanet/server5/msg-sun08/config</i>.
6. <b>DOMAINS</b> - specify the names of the domains hosted in the current messaging server instance. Multiple domains can be provided as a comma-separated list, but ensure that there is no space between a comma and a domain name. Example: <i>chn.egurkha.com,eg.egurkha.com</i>. Only the users present in the specified domains will be monitored.
7. <b>DETAILED DIAGNOSIS</b> - To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the <b>On</b> option. To disable the capability, click on the <b>Off</b> option.
The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:
➢ The eG manager license should allow the detailed diagnosis capability
➢ Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.</td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for every domain</td></tr>
<tr><td rowspan="3"><b>Measurements made by the test</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
<tr><td><b>Users connected:</b>
Indicates the number of user accounts present in a domain.</td><td>Number</td><td></td></tr>
<tr><td><b>Disk space consumed:</b>
Indicates the total disk space consumed by the messages in the user accounts of a particular domain.</td><td>MB</td><td>If this value is high then it indicates that the users of this domain are consuming more disk space. The detailed diagnosis of this measure, if enabled, provides the list of top 10 users of a domain who have consumed maximum disk resources. This information helps users in identifying the user accounts which utilize more space, and take the necessary measures to free adequate disk space in those accounts</td></tr>
</table>

	Messages stored: Indicates the total number of messages present in the user accounts of a particular domain	Number	The detailed diagnosis of this measure, if enabled, provides the list of top 10 users of a domain holding maximum number of messages in their respective user accounts.

`

6

Monitoring Tibco EMS Servers

TIBCO Enterprise Message Service (EMS) is fully compliant Java Message Service (JMS) implementation from TIBCO with some enterprise-class enhancements. The Java Message Service makes it easy to write business applications that asynchronously send and receive critical business data and events.

Issues in the performance of the Tibco EMS can therefore obstruct the exchange of critical data across the environment, stalling key business transactions in the process and causing significant loss of revenue and reputation. It is therefore imperative to monitor the Tibco EMS.

eG Enterprise offers a dedicated *Tibco EMS* monitoring model that continuously monitors the Tibco EMS, and proactively alerts administrators to potential performance bottlenecks.

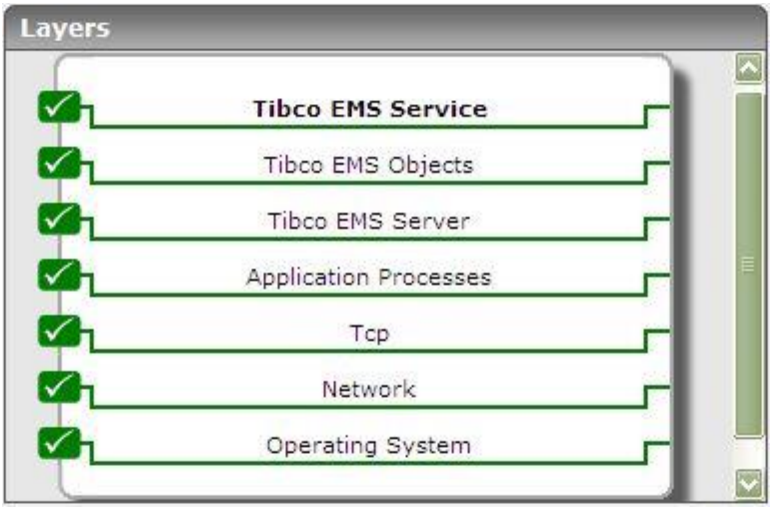


Figure 6.1: The layer model of the Tibco EMS Server

Using the metrics reported by this model, the following questions can be accurately answered:

- What is the current mode of operation of the server - active, inactive, or fault-tolerant standby mode?
- What is the total queue and topic load imposed by applications on the EMS Server? What type of queues and topics are maximum on the server?
- Are too many connections to the server idle?
- How busy is the EMS server in terms of the number of sessions it is handling?
- Are there any messages for durable subscriptions on the server?
- How many applications are sending messages and retrieving messages from the server?
- Are there too many pending messages on the server? Which queue/topic has the maximum number of pending messages?
- Are the pending messages too heavy? Which queue/topic contains the heaviest pending messages?
- Is there very little free message memory on the server?
- Which queue/topic on the server is static? Which queues/topics are dynamic or temporary?
- Which user to the server has the maximum number of sessions and connections open on the server?
- Are any durable subscribers offline currently?
- Are there too messages awaiting delivery to any durable subscriber?
- The heaviest pending message is meant for which durable subscriber?
- Has the log file utilized its maximum allocated space?

The sections that will follow discuss each of the top-3 layers of Figure 6.1, as the other layers have already been discussed in other documents.

6.1 The Tibco EMS Server Layer

The tests mapped to this layer reveal the overall health of the Tibco EMS server, and promptly alert administrators to the following problem conditions:

a. Excessive space usage by the server log files;

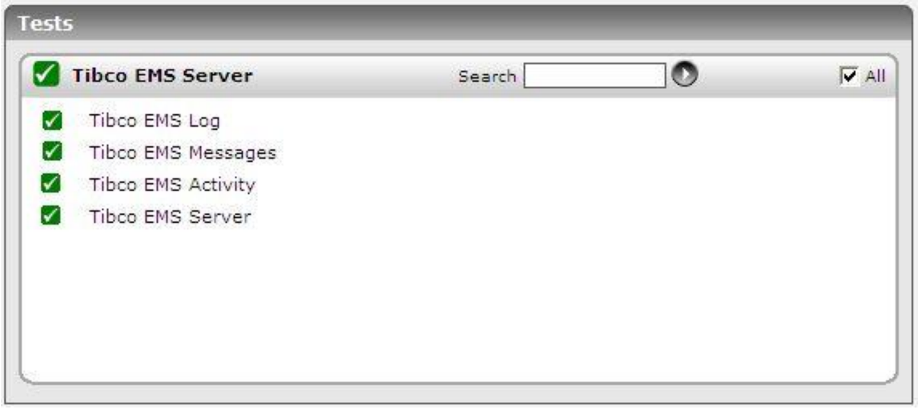
b. A large number of pending messages on the server;

c. Excessive usage of message memory;

d. Too many idle connections on the server;

e. A server overload caused by a large number of topics/queues on the server;

f. Non-availability of the server



☑ Tibco EMS Log
☑ Tibco EMS Messages
☑ Tibco EMS Activity
☑ Tibco EMS Server

Figure 6.2: The tests mapped to the Tibco EMS Server Layer

6.1.1 Tibco EMS Log

The EMS server can be configured to write a variety of information to a log file. Server activities can be efficiently tracked using the information logged in the log files. Several parameters and commands control the location, logged information, and general configuration of the log files. One such parameter is the *logfile_max_size* configuration parameter that governs the maximum size upto which a log file can grow. To make sure that the log file does not grow boundlessly, this test periodically monitors the size of the log file and promptly alerts administrators if the log file is about to exceed the size limit set.

Purpose	Periodically monitors the size of the log file and promptly alerts administrators if the log file is about to exceed the size limit set
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** - The host for which the test is being configured
3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:

```
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show server
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show durables
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show queues
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show topics
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show connections
```

For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:

```
tcp://192.168.10.28:9090 show server
```

The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.

Outputs of the test

One set of results for Tibco EMS server being monitored

Measurements made by the test

Measurement	Measurement Unit	Interpretation
Log file size: Indicates the current size of the log file.	KB	
Max log file size: Indicates the maximum allocated size of the log file.	KB	This measure typically returns the value set for the *logfile_max_size* configuration parameter.

	Log file usage: Indicates the percentage of allocated space that is currently used by the log file.	Percent	If the value of this measure is 100%, it indicates that the *logfile_max_size* threshold has been violated - i.e., the log file has reached its maximum size. In this case, the contents of the log file are copied to a file with the same name as the current log file, except that a sequence number is appended to the name of the backup file. The server queries the directory and determines the first available sequence number. For example, if the current log file is named tibems.log, the first copy is named tibems.log.1, the second is named tibems.log.2, and so on. To further regulare the space usage of the log files, you can also dynamically force the log file to be backed up and truncated using the *rotatelog* command in *tibemsadmin*.

6.1.2 Tibco EMS Messages

Messages are structured data that one application sends to another. Typically, the messages sent by an application are either delivered to a queue on the EMS server or published to a topic on the EMS server. In addition, you may have one/more messages that just wait on the EMS server pending delivery to a queue or a topic - this could be because of the messages being too large in size or the queue/topic being too crowded already to accomodate more messages.

This test takes stock of all the messages on an EMS server, regardless of where they are (whether queue/topic/pending delivery to queue or topic), and reports a wealth of performance statistics pertaining to the memory and storage space used by these messages on the server.

Purpose	Takes stock of all the messages on an EMS server, regardless of where they are (whether queue/topic/pending delivery to queue or topic), and reports a wealth of performance statistics pertaining to the memory and storage space used by these messages on the server
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed
	2. **HOST** - The host for which the test is being configured
	3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
	4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show server`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show durables`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show queues`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show topics`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show connections`
	For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:
	`tcp://192.168.10.28:9090 show server`
	The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.		

Outputs of the test	One set of results for the Tibco EMS server being monitored		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Pending messages:** Indicates the number of pending messages on the server.	Number	If a consistent and significant increase is noticed in the value of this measure over a period of time, it could indicate either or both the following: g. A number of messages are still waiting to be delivered to queues or topics; h. Consumers are not / are unable to retrieve many messages from the queues/topics If this situation continues, it can unduly overload the server, forcing it to stall message delivery and return errors to the producers; this in turn may cause critical messages to not reach consumers on time. In this case therefore, the bottleneck areas must be identified and resolved quickly.

	Pending message size: Indicates the total size of all the pending messages on the server.	KB	Ideally, the value of this measure should be low. While sporadic spikes in the value are normal and may indicate the entry of one/more bulky messages, a steady increase in the value of this measure over time could be a cause for concern, as it may induce the excessive consumption of the storage capacity of the server. This space erosion can adversely impact the overall server performance. To avoid such adversities, you might want to consider fine-tuning the EMS server to handle pending messages better. For instance, you can use the *maxBytes* configuration parameter of a queue/topic. *maxbytes* defines the maximum size (in bytes) of all messages that can be waiting in a queue or waiting to be delivered to the durable subscribers of a topic. By reducing the *maxBytes* of a queue/topic, you can reduce the size of all messages held by that destination, thus significantly reducing the resource foot-print of the messages within. If this limit is violated, the server returns an error to the producer. Similarly, you can try enabling 'Flow control'. Flow control is a feature that controls the flow of messages to a destination. If this capability is enabled for the server, then, you can use the *flowControl* configuration parameter to configure each destination with a target maximum size for storing pending messages. If need be, you can reduce the *flowControl* value, so that the storage capacity utilized by messages waiting to be delivered to a destination is reduced. If this limit is violated, the server blocks producers from sending any more messages.

	Free message memory: Indicates the amount of storage memory unused on the server.	KB	Adequate storage memory should always be available on the server for storing messages. Excessive memory usage over time can exhaust the storage capacity of the server - the lack of sufficient memory can slowdown and can even bring to a halt, all server operations. To avoid such adversities, you can regulate the space usage of the messages on the server. For instance, you can set the *maxBytes* threshold for a queue/topic. *maxbytes* defines the maximum size (in bytes) of all messages that can be waiting in a queue or waiting to be published to a topic. By reducing the *maxBytes* of a queue/topic, you can reduce the size of all messages held by that destination, thus significantly reducing the resource foot-print of the messages within. If this limit is violated, the server returns an error to the producer. Similarly, you can try enabling 'Flow control'. Flow control is a feature that controls the flow of messages to a destination. If this capability is enabled for the server, then, you can use the *flowControl* configuration parameter to configure each destination with a target maximum size for storing pending messages. If need be, you can reduce the *flowControl* value, so that the storage capacity utilized by messages waiting to be delivered to a destination is reduced. If this limit is violated, the server blocks producers from sending any more messages.
	Used message memory: Indicates the amount of storage memory that is currently in use for storing messages on the server.	KB	
	Total message memory: Indicates the total memory allocated for storing messages on the server.	KB	
	Message memory used: Indicates the percentage of memory used for storing messages.	Percent	
	Message memory pooled: Indicates the size of the pools of storage allocated for messages.	KB	
	Synchronous storage: Indicates the size of the synchronous storage memory.	KB	A synchronous storage memory is the one which can store only one message at a time.

	Asynchronous storage: Indicates the size of the asynchronous storage memory.	KB	A asynchronous storage memory is the one which allows multiple message storing at a time.
	Fsync: A asynchronous storage memory is the one which allows multiple message storing at a time.	Number	Critical message transaction will be done successfully by enabling the "fail safe" configuration for a specific queue. By this the messages for a "fail safe" queue are synchronously written to the disk. By enabling this, the message transactions will not be lost even when the server goes down. In other words, this mode is decribed as "reliable mode". The value 0 for this measure indicates that the 'fail safe' mode is disabled and and the value 100 for this measure indicates the 'fail safe' is enabled.
	Inbound messages: Indicates the number of incoming messages to the message memory.	Number	
	Outbound messages: Indicates the number of outgoing messages from the message memory.	Number	

6.1.3 Tibco EMS Activity Test

This test provides a snapshot of the level of activity on the server by reporting the number of sessions on and applications connecting to the server. Besides revealing how busy the server is, this test helps isolate idle connections to the server so that such connections can be promptly removed.

Purpose	This test provides a snapshot of the level of activity on the server by reporting the number of sessions on and applications connecting to the server. Besides revealing how busy the server is, this test helps isolate idle connections to the server so that such connections can be promptly removed.
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** - The host for which the test is being configured
3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:

```
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show server
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show durables
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show queues
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show topics
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show connections
```

For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:

```
tcp://192.168.10.28:9090 show server
```

The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.

Outputs of the test

One set of results for the Tibco EMS server being monitored

Measurements made by the test

Measurement	Measurement Unit	Interpretation
Used connections: Indicates the number of virtual connections to the server that are actively used.	Number	A Connection object encapsulates a virtual connection with the server. A connection typically represents a communication link between the application and the messaging server. A connection is a fairly heavyweight object, and therefore most clients will use one connection for all messaging. You may create multiple connections, if needed by your application. The value of this measure therefore will indicate whether any applications require multiple connections to the EMS server.

	Connections used percent: Indicates the percentage of total connections to the server that are in use.	Percent	Ideally, this value should be high. A low ratio of used connections indicates that too many connections to the server are currently unused - i.e., are idle. Unused open connections are eventually closed, but they do consume resources that could be used for other applications. Too many idle connections therefore can accelerate the resource drain on the EMS server, thereby compelling other applications to contend for limited resources. A low value for this measure is hence a cause for concern.
	Sessions: Indicates the number of sessions on the server.	Number	A session represents a single-threaded context for sending and receiving messages. A session is single-threaded so that messages are serialized, meaning that messages are received one-by-one in the order sent. The benefit of a session is that it supports transactions. If the user selects transaction support, the session context holds a group of messages until the transaction is committed, then delivers the messages. Before committing the transaction, the user can cancel the messages using a rollback operation. A session allows users to create message producers to send messages, and message consumers to receive messages. This measure is generally an indicator of how busy the EMS server is.
	Producers: Indicates the number of producers communicating with the server.	Number	Messages are structured data that one application sends to another. The creator of a message is known as a producer. Using the value reported by this measure, you can accurately figure out how many applications are sending messages to the queues on the server.
	Consumers: Indicates the number of consumers communicating with the server.	Number	Messages are structured data that one application sends to another. The receiver of messages is known as a consumer. Using the value reported by this measure, you can accurately figure out how many applications are retrieving messages from the queues on the server.

	Durables: Indicates the number of messages for the durable subscriptions.	Number	Each message consumer subscribes to a topic. When a message is published to that topic, all subscribed consumers receive the message. By default, subscribers only receive messages when they are active. If the messages are delivered when the subscriber is not available, the subscriber does not receive these messages. Java Message Service(JMS) specifies a way to remove the part of timing dependency by allowing subscribers to create durable subscriptions. Messages for durable subscriptions are stored on the server until the message expires or the storage limit is reached. Subscribers can receive messages from a durable subscription even if the subscriber was not available when the message was originally delivered.

6.1.4 Tibco EMS Server Test

This test reports the current state of the server, and also indicates the load on the server in terms of the number and type of topics and queues handled by it.

Purpose	Reports the current state of the server, and also indicates the load on the server in terms of the number and type of topics and queues handled by it
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** - The host for which the test is being configured
3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:

```
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show server
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show durables
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show queues
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show topics
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show connections
```

For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:

```
tcp://192.168.10.28:9090 show server
```

The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.

Outputs of the test

One set of results for the Tibco EMS server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Is active?: Indicates the current state of the EMS Server.	Number	The value 0 for this measure indicates that the server being monitored is inactive, and the value 2 indicates that it is active. The value 1 on the other hand indicates that the server is currently in the fault tolerant standby mode. The fault-tolerant state is operated by configuring a pair of servers - namely, primary and backup. The primary and backup servers act as a pair, with the primary server accepting the client connections and performing the work of handling messages, and the secondary server acting as a backup in case of failure. If the primary server fails, the backup server resumes operation in its place.

	Total topics: Indicates the total number of topics available on the server.	Number	Topics is a distribution mechanism for publishing messages that are delivered to multiple subscribers.The publisher generally addresses messages to a topic. Many publishers can publish to the same topic, and a message from a single publisher can be received by many subscribers. Subscribers subcribe to topics and all messages published to the topic are received by all subscribers to the topic. The value reported by this measure includes dynamic, static and temporary topics, and thus indicates the total topic load on the server.
	Dynamic topics: Indicates the total number of dynamic topics on the server.	Number	Destinations for message can be either queues or topics, and these destinations can either be created statically or dynamically. Dynamic topics do not appear in the configuration files and exist as long as there are messages or consumers on the destination. Since these topics are created dynamically by applications, this measure serves as a good indicator of the topic load imposed by applications on the EMS server.

	Temporary topics: Indicates the number of temporary topics on the server.	Number	Temporary destinations (temporary queues or temporary topics) are proposed as a lightweight alternative in a scalable system architecture that could be used as unique destinations for replies. Such destinations have a scope limited to the connection that created it, and are removed on the server side as soon as the connection is closed. Owing to their short life span, these topics may not have a lasting effect on the load/overall performance of the EMS server. For the same reason and those listed below, the temporary topics are not ideal destinations for messages: i. A temporary destination can only be consumed by the connection that created it. j. When you close the connection that has a temporary destination, the destination is closed and its contents are lost. k. You cannot have durable subscriptions to a TemporaryTopic. l. Each temporary destination is unique and cannot be copied. m. Temporary destinations cannot be routed using an enterprise messaging service. Typically, the value of this measure will be low. However, if the value appears to significantly increase with time, it could indicate a bottleneck in the closure of connections; further investigation may be required in this regard.
	Static topics: Indicates the number of static topics on the server..	Number	The destination for a message can be either created statically in the server configuration files or dynamically by a client application. Static topics are those topics for which configuration information resides within configuration files for the EMS server. These topics are typically administered by the server itself. The value of this measure therefore indicates the server workload in terms of the number of static topics it is currently administering.

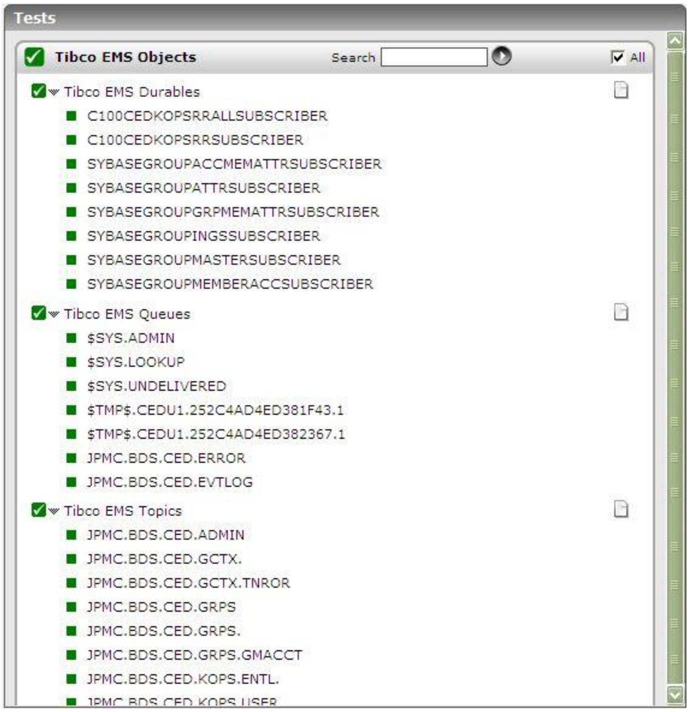
	Total queues: Indicates the total number of queues..	Number	The point-to-point style of messaging uses a queue to store messages until they are received. The message producer sends the message to the queue; the message consumer retrieves messages from the queue and sends acknowledgement that the message was received. Based on how it is created, a queue can be classified as dynamic, static, or temporary. Based on what its created for, a queue can be configured to be 'exclusive' or 'non-exclusive'. If a queue is exclusive, then all messages in this queue can only be retrieved by the first consumer specified for the queue. Exclusive queues are useful when you want only one application to receive messages for a specific queue. If the queue is not exclusive, any number of receivers can retrieve messages from the queue. Non-exclusive queues are useful for balancing the load of incoming messages across multiple receivers. The value of this measure includes queues of all the above-mentioned types, and is useful for determining the load on the server in terms of the number of queues it handles.
	Dynamic queues: Indicates the total number dynamic queues that are created.	Number	Dynamic queues do not appear in the configuration files and exist as long as there are messages or consumers on the destination. Since these queues are created dynamically by applications, this measure serves as a good indicator of the queue load imposed by applications on the EMS server.

	Temporary queues: Indicates the number of temporary queues on the server.	Number	Temporary destinations (temporary queues or temporary topics) are proposed as a lightweight alternative in a scalable system architecture that could be used as unique destinations for replies. Such destinations have a scope limited to the connection that created it, and are removed on the server side as soon as the connection is closed. Owing to their short life span, temporary queues may not have a lasting effect on the load/overall performance of the EMS server. For the same reason and those listed below, the temporary queues are not ideal destinations for messages: n. A temporary destination can only be consumed by the connection that created it. o. When you close the connection that has a temporary destination, the destination is closed and its contents are lost. You cannot have durable subscriptions to a TemporaryTopic. p. Each temporary destination is unique and cannot be copied. q. Temporary destinations cannot be routed using an enterprise messaging service. Typically, the value of this measure will be low. However, if the value appears to significantly increase with time, it could indicate a bottleneck in the closure of connections; further investigation may be required in this regard.

	Static queues: Indicates the number of static queues on the server.	Number	The destination for a message can be either created statically in the server configuration files or dynamically by a client application. Static queues are those queues for which configuration information resides within configuration files for the EMS server. These queues are typically administered by the server itself. The value of this measure therefore indicates the server workload in terms of the number of static queues it is currently managing.

6.2 The Tibco EMS Objects Layer

This layer zooms into the performance of each durable subscriber, queue, and topic on the server, and reports which subscriber/queue/topic has the maximum number of pending messages on the server, or is utilizing too much message memory.



- Tibco EMS Durables
  - C100CEDKOPSRRALLSUBSCRIBER
  - C100CEDKOPSRRSUBSCRIBER
  - SYBASEGROUPACCMEMATTRSUBSCRIBER
  - SYBASEGROUPATTRSUBSCRIBER
  - SYBASEGROUPGRPMEMATTRSUBSCRIBER
  - SYBASEGROUPINGSSUBSCRIBER
  - SYBASEGROUPMASTERSUBSCRIBER
  - SYBASEGROUPMEMBERACCSUBSCRIBER
- Tibco EMS Queues
  - $SYS.ADMIN
  - $SYS.LOOKUP
  - $SYS.UNDELIVERED
  - TMP.CEDU1.252C4AD4ED381F43.1
  - TMP.CEDU1.252C4AD4ED382367.1
  - JPMC.BDS.CED.ERROR
  - JPMC.BDS.CED.EVTLOG
- Tibco EMS Topics
  - JPMC.BDS.CED.ADMIN
  - JPMC.BDS.CED.GCTX.
  - JPMC.BDS.CED.GCTX.TNROR
  - JPMC.BDS.CED.GRPS
  - JPMC.BDS.CED.GRPS.
  - JPMC.BDS.CED.GRPS.GMACCT
  - JPMC.BDS.CED.KOPS.ENTL.
  - JPMC.BDS.CED.KOPS.USER

Figure 6.3: The tets mapped to the Tibco EMS Objects

6.2.1 Tibco EMS Durables Test

Generally, each message consumer subscribes to a topic. When a message is published to that topic, all subscribed consumers receive the message. Because of this there might be time dependency. By default, subscibers only receive messages when they are active. If the messages are delivered when the subscriber is not available, the subscriber does not receive these messages.

Java Message Service(JMS) specifies a way to remove the part of timing dependency by allowing subscribers to create durable subscriptions. Messages for durable subscriptions are stored on the server until the message expires or the storage limit is reached. Subscribers can receive messages from a durable subscription even if the subscriber was not available when the message was originally delivered.

When an application restarts and recreates a durable subscriber with the same ID, all messages stored on the server for that topic are published to the durable subscriber.

Until such time, the pending messages will remain on the EMS server, consuming machine resources. If the messages are allowed to grow limitlessly in size and number over time, they can drain the server of its resources, thereby choking critical server operations. Using this test, you can continuously monitor the status, activities, and pending messages of durable subscribers, and be proactively alerted if the pending messages appear to be growing significantly in size and/or number. Besides enabling you to control message growth, the metrics reported by this test also enable you to avert impending resource shortages on the server.

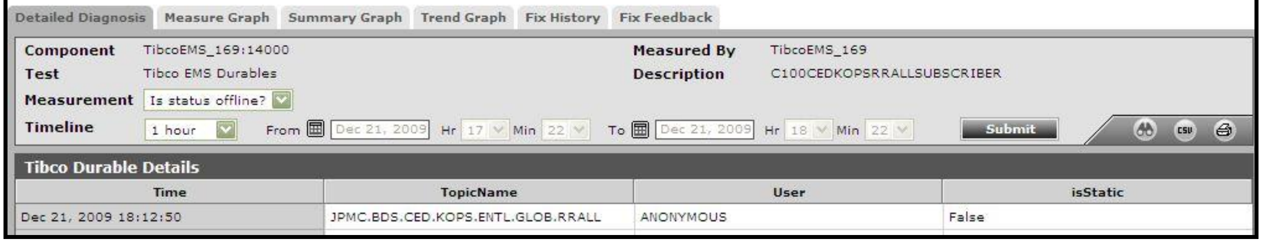
Purpose	Continuously monitors the status, activities, and pending messages of durable subscribers, and proactively alerts if the pending messages appear to be growing significantly in size and/or number.
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed
	2. **HOST** - The host for which the test is being configured
	3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
	4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show server`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show durables`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show queues`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show topics`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show connections`
	For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:
	`tcp://192.168.10.28:9090 show server`
	The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.
	5. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise system embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option against **DETAILED DIAGNOSIS**. To disable the capability, click on the **Off** option.
	The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:
	r. The eG manager license should allow the detailed diagnosis capability
	s. Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.		

Outputs of the test	One set of results for each durable subscriber on the Tibco EMS server being monitored		
Measurements made by the	**Measurement**	**Measurement Unit**	**Interpretation**

test	**Is status offline?:** Indicates the current state of this durable subscriber.	Boolean	If the value of this measure is 0, it indicates that the subscriber is offline, and if the value is 1, it indicates that the subscriber is online. Use the detailed diagnosis of this measure to know the topics to which the durable subscriber is subscribing.
	Pending messages: Indicates the number of pending messages for this durable subscriber.	Number	If the value of this measure increases significantly over time, it could mean that more messages are still waiting to be delivered to subscribers. This could primarily be because, while the publishers are sending messages to the topic quickly, the subscriber is not receiving the messages quiet as quickly. One of the reasons for this slowdown could be a network connectivity issue between the subscriber and the EMS server - a poor network link could be delaying message delivery at one end, while on the other end, the topic may be experiencing a major influx of messages from the publisher. The pending message count can also increase, if the subscriber has been offline for a long time. During this period of inactivity, existing messages will remain undelivered on the server, and more messages may continue to be published to the topic by the publisher, thereby increasing the pending message count. However, regardless of the reason, the number of pending messages should be kept under control at all times, as they consume considerable server resources.

	Message size: Indicates the total size of pending messages for this durable subscriber.	KB	If the size grows significantly over time, it could indicate that messages are getting published on the topic but are not being delivered to subscribers quiet as quickly. As already mentioned, the slowdown in delivery can be attributed to the durable subscriber being offline for a long time. Practical issues - such as bad network links - faced by subscribers while receiving messages from the topic can also delay/halt delivery. Regardless of the root cause, the message size should not be allowed to grow uncontrollably, as it may cause a serious resource contention on the server. To limit the growth of pending messages on a topic, use the *maxBytes* configuration for that topic. For topics, *maxBytes* limits the total size (in bytes) of all messages waiting for delivery to each durable subscriber on that topic. If this limit is violated, then messages will be go undelivered, thus causing the receivers to lose critical data.



Component TibcoEMS_169:14000

Test Tibco EMS Durables

Measurement Is status offline?

Timeline 1 hour From Dec 21, 2009 Hr 17 Min 22 To Dec 21, 2009 Hr 18 Min 22

Measured By TibcoEMS_169

Description C100CEDKOPSRRALLSUBSCRIBER

Tibco Durable Details

Time	TopicName	User	isStatic
Dec 21, 2009 18:12:50	JPMC.BDS.CED.KOPS.ENTL.GLOB.RRALL	ANONYMOUS	False

Figure 6.4: The detailed diagnosis of the Is status offline? measure

6.2.2 Tibco EMS Queues Test

The point-to-point style of messaging uses a queue to store messages until they are received. The message producer sends the message to the queue; the message consumer retrieves messages from the queue and sends acknowledgement that the message was received. More than one producer can send messages to the same queue, and more than one consumer can retrieve messages from the same queue. The queue can be configured to be exclusive, if desired. If the queue is exclusive, then all queue messages can only be retrieved by the first consumer specified for the queue. Exclusive queues are useful when you want only one application to receive messages for a specific queue. If the queue is not exclusive, any number of receivers can retrieve messages from the queue. Non-exclusive queues are useful for balancing the load of incoming messages across multiple receivers. Regardless of whether the queue is exclusive or not, only one consumer can ever retrieve each message that is placed on the queue.

For every queue configured on the EMS server, this test enables you to track the length of the queue and its size (in bytes), so that you can be promptly alerted to any abnormal increase in queue length or to any load imbalances experienced by the queue. In addition, the test reports the queue type and the number of receivers to the messages in the queue.

<table>
<tr><td><b>Purpose</b></td><td colspan="3">For every queue configured on the EMS server, this test enables you to track the length of the queue and its size (in bytes), so that you can be promptly alerted to any abnormal increase in queue length or to any load imbalances experienced by the queue</td></tr>
<tr><td><b>Target of the test</b></td><td colspan="3">A Tibco EMS Server</td></tr>
<tr><td><b>Agent deploying the test</b></td><td colspan="3">An internal agent</td></tr>
<tr><td><b>Configurable parameters for the test</b></td><td colspan="3">1. <b>TEST PERIOD</b> – How often should the test be executed
2. <b>HOST</b> - The host for which the test is being configured
3. <b>PORT –</b> The SMTP port of the iPlanet/SunONE messaging server
4. <b>COMMANDPATH</b> – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:<pre>tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show server
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show durables
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show queues
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show topics
tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
show connections</pre>For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:<pre>tcp://192.168.10.28:9090 show server</pre>The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the <b>COMMANDPATH</b> text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.
5. <b>IGNORE QUEUES</b> - Specify a comma-separated list of queues to be excluded from monitoring in the <b>IGNORE QUEUES</b> text box. A * can be used in the queue name to indicate leading or trailing spaces. For instance, to ignore queues with names that embed the string <i>sys</i>, your <b>IGNORE QUEUES</b> specification can be: <b>*sys*</b></td></tr>
<tr><td><b>Outputs of the test</b></td><td colspan="3">One set of results for each queue on the Tibco EMS server being monitored</td></tr>
<tr><td><b>Measurements made by the</b></td><td><b>Measurement</b></td><td><b>Measurement Unit</b></td><td><b>Interpretation</b></td></tr>
</table>

test	**Receivers:** Indicates the number of currently active receivers in this queue.	Number	Receivers can retrieve messages from the queue. By default receivers can receive multiple messages at the same instant. You can also configure the receivers to receive only one message at a time. The value of this measure will help you ascertain the workload of the queue in terms of the number of receivers it supports.
	Pending messages: Indicates the number of pending messages in this queue.	Number	If the value of this measure increases significantly over time, it could mean that the messages are getting accumulated in the queue. This could primarily be because the producers are sending messages more rapidly than the speed of consumption by the consumers. One of the reasons for this slowdown could be that the consumer has been offline for a long time - this also allows more messages to be enqueued. Connectivity issues faced by the consumer while retrieving messages from the queue could also delay consumption. Regardless of the reason, the queue length should be kept under a check at all times, so as to conserve server resources and the loss of critical messages.
	Message size: Indicates the total size of pending messages in this queue.	KB	If the size grows significantly over time, it could indicate that messages are getting added to the queue but are not being consumed quiet as quickly. As already mentioned, the slowdown in consumption can be attributed to the consumer being offline for a long time. Practical issues - such as bad network links - faced by consumers while retrieving messages from the queue can also delay/halt consumption. Regardless of the root cause, the message size should not be allowed to grow uncontrollably, as it may completley erode the storage resources of the server. To limit the growth of a queue, use the the *maxBytes* configuration for that queue. *maxbytes* defines the maximum size (in bytes) of all messages that can be waiting in a queue. If this limit is violated, an error is returned to the producers.

	IsStatic: Indicates whether the queue is static or not.	Boolean	The value 0 for this measure indicates that the queue is static, and the value 1 indicates that it is non-static - in this case, the queue can be of type dynamic or temporary. A static queue is typically created statically in the server configuration files, and the dynamic queue is created on-the-fly by applications. The dynamic queues do not appear in the configuration files, and exist as long as there are messages or consumers on the destination. You can identify dynamic queues in the administration tool using the asterisk (*) that will pre-fix their names. Temporary destinations (temporary queues or temporary topics) are proposed as a lightweight alternative in a scalable system architecture that could be used as unique destinations for replies. Such destinations have a scope limited to the connection that created it, and are removed on the server side as soon as the connection is closed. Owing to their short life span, these queues may not have a lasting effect on the load/overall performance of the EMS server.

6.2.3 Tibco EMS Topics

Topics are a distribution mechanism for publishing messages that are delivered to multiple subscribers. A topic's properties are set when the destination is created. The publisher generally addresses messages to a topic. Many publishers can publish to the same topic, and a message from a single publisher can be received by many subscribers. Subscribers subcribe to topics and all messages published to the topic are received by all subscribers to the topic.

This test allows you to keep tabs on the number and size of pending messages to each topic, so that you can accurately identify topics to which too many messages are pending and investigate the reasons for the same. In addition, the test reports the topic type and the number of subscribers to a topic, so that you can easily determine how popular the topic is.

Purpose	This test allows you to keep tabs on the number and size of pending messages to each topic, so that you can accurately identify topics to which too many messages are pending and investigate the reasons for the same.
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test

1. **TEST PERIOD** – How often should the test be executed
2. **HOST** - The host for which the test is being configured
3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:

   ```
   tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
   show server
   tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
   show durables
   tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
   show queues
   tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
   show topics
   tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS}
   show connections
   ```

   For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:

   ```
   tcp://192.168.10.28:9090 show server
   ```

   The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.
5. **IGNORE TOPICS** - Specify a comma-separated list of topics to be excluded from monitoring in the **IGNORE TOPICS** text box. A * can be used in the topic name to indicate leading or trailing spaces. For instance, to ignore topics with names that embed the string *sys*, your **IGNORE TOPICS** specification can be: **sys**

Outputs of the test

One set of results for each topic on the Tibco EMS server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Subscribers: Indicates the number of current subscribers to this topic.	Number	The value reported by this measure includes the number of durable subscribers as well. This measure therefore is a good indicator of how popular a particular topic is.
	Durables: Indicates the number of durable subscribers to this topic.	Number	

	Pending messages: Indicates the number of messages on this topic waiting to be delivered to subscribers.	Number	If the value of this measure increases significantly over time, it could mean that more messages are still waiting to be delivered to subscribers. This could primarily be because, while the publishers are sending messages to the topic quickly, the subscriber is not receiving the messages quiet as quickly. One of the reasons for this slowdown could be a network connectivity issue between the subscriber and the EMS server - a poor network link could be delaying message delivery at one end, while on the other end, the topic may be experiencing a major influx of messages from the publisher. The pending message count can also increase, if a topic has durable subscribers, and one/more of these subscriber have been offline for a long time. During this period of inactivity, existing messages will remain undelivered on the server, and more messages may continue to be published to the topic by the publisher, thereby increasing the pending message count. However, regardless of the reason, the number of pending messages should be kept under control at all times, as they consume considerable server resources.
	Messages size: Indicates the total size of pending messages.	KB	If the size grows significantly over time, it could indicate that messages are getting published on the topic but are not being delivered to subscribers quiet as quickly. As already mentioned, the slowdown in delivery can be attributed to the durable subscriber being offline for a long time. Practical issues - such as bad network links - faced by subscribers while receiving messages from the topic can also delay/halt delivery. Regardless of the root cause, the message size should not be allowed to grow uncontrollably, as it may cause a serious resource contention on the server. To limit the growth of pending messages on a topic, use the *maxBytes* configuration for that topic. For topics, *maxBytes* limits the total size (in bytes) of all messages waiting for delivery to each durable subscriber on that topic. If this limit is violated, then messages will be go undelivered, thus causing the receivers to lose critical data.

	IsStatic: Indicates whether the topic is static or not.	Boolean	The value 0 for this measure indicates that the topic is in static state and the value 1 for this measure indicates that it is in a non-static (i.e., dynamic or temporary) state. A static topic is typically created statically in the server configuration files, and the dynamic topic is created on-the-fly by applications. The dynamic topics do not appear in the configuration files, and exist as long as there are messages or consumers on the destination. You can identify dynamic topics in the administration tool using the asterisk (*) that will pre-fix their names. Servers connected by routes exchange messages sent to temporary topics. As a result, temporary topics are ideal destinations for reply messages in request/reply interactions.

6.3 The Tibco EMS Service Layer

The test mapped to this layer focus on the user activity on the server, and reveals which user is having the maximum number of sessions and connections open on the server.

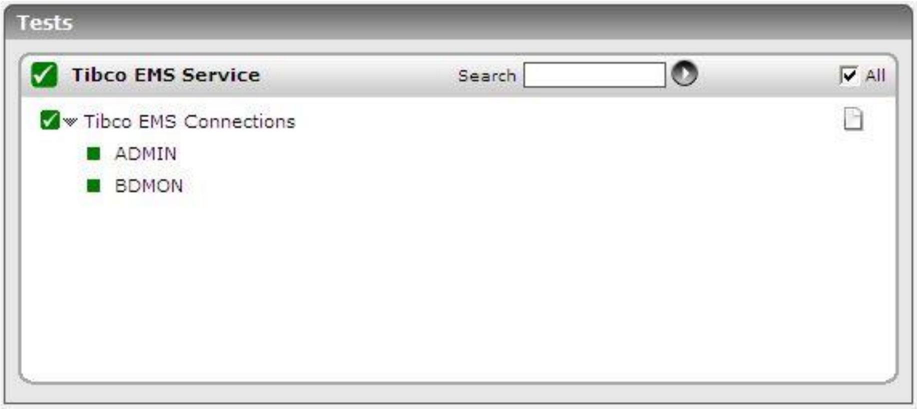


Figure 6.5: The tests mapped to the Tibco EMS Service layer

6.3.1 Tibco EMS Connections Test

This test monitors the user activity on the EMS server, and reports the number of connections and sessions initiated by each user on the server. The users with the maximum number of open sessions on the server can thus be quickly identified and their activities closely tracked.

Purpose	Monitors the user activity on the EMS server, and reports the number of connections and sessions initiated by each user on the server

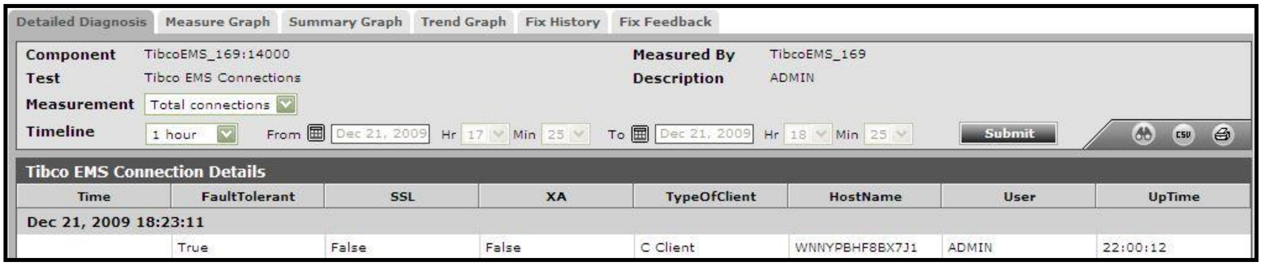
Target of the test	A Tibco EMS Server
Agent deploying the test	An internal agent

Configurable parameters for the test	1. **TEST PERIOD** – How often should the test be executed
	2. **HOST** - The host for which the test is being configured
	3. **PORT –** The SMTP port of the iPlanet/SunONE messaging server
	4. **COMMANDPATH** – Prior to monitoring the Tibco EMS server, you will have to build a .bat or .sh file (depending upon the operating system on which Tibco EMS is functioning) bundled with the commands that the eG agent needs to execute on the Tibco EMS server for collecting the required metrics. The commands to be invoked by the the .bat or .sh file are as follows:
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show server`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show durables`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show queues`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show topics`
	`tcp://{IPAddressorHostName_of_TibcoEMS}:{PortNumber_of_TibcoEMS} show connections`
	For instance, if the IP address of your Tibco EMS server is 192.168.10.28 and its port is say, 9090, then a sample command in the .bat or .sh file would be:
	`tcp://192.168.10.28:9090 show server`
	The .bat/.sh file so created can be saved to any location on the Tibco EMS host. Then, while configuring this test, make sure you provide the full path to this .bat or .sh file in the **COMMANDPATH** text box so that, the agent can execute the file, invoke the commands bundled into it, and extract the desired metrics from the server.
	5. **DETAILED DIAGNOSIS** - To make diagnosis more efficient and accurate, the eG Enterprise system embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option against **DETAILED DIAGNOSIS**. To disable the capability, click on the **Off** option.
	The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled:
	t. The eG manager license should allow the detailed diagnosis capability
	u. Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Outputs of the test	One set of results for each user connected to the Tibco EMS server being monitored

Measurements made by the test	Measurement	Measurement Unit	Interpretation
	Total connections: Indicates the total number of connections on the server for this user.	Number	The detailed diagnosis of this measure, if enabled, will provide the details of each connection that this user has established with the EMS server. The host from which the user is connecting and the total uptime of the connection can be determined using this information.
	Total sessions Indicates the total number of sessions on the server for this user.	Number	

The detailed diagnosis of the *Total connections* measure, if enabled, will provide the details of each connection that this user has established with the EMS server. The host from which the user is connecting and the total uptime of the connection can be determined using this information.



Component TibcoEMS_169:14000
Test Tibco EMS Connections
Measured By TibcoEMS_169
Description ADMIN
Measurement Total connections
Timeline 1 hour From Dec 21, 2009 Hr 17 Min 25 To Dec 21, 2009 Hr 18 Min 25

Tibco EMS Connection Details

Time	FaultTolerant	SSL	XA	TypeOfClient	HostName	User	UpTime
Dec 21, 2009 18:23:11							
	True	False	False	C Client	WNNYPBHF8BX7J1	ADMIN	22:00:12

Figure 6.6: The detailed diagnosis of the Total connections measure

7

Conclusion

This document has described in detail the monitoring paradigm used and the measurement capabilities of the eG Enterprise suite of products with respect to **messaging servers**. For details of how to administer and use the eG Enterprise suite of products, refer to the user manuals.

We will be adding new measurement capabilities into the future versions of the eG Enterprise suite. If you can identify new capabilities that you would like us to incorporate in the eG Enterprise suite of products, please contact support@eginnovations.com. We look forward to your support and cooperation. Any feedback regarding this manual or any other aspects of the eG Enterprise suite can be forwarded to feedback@eginnovations.com.