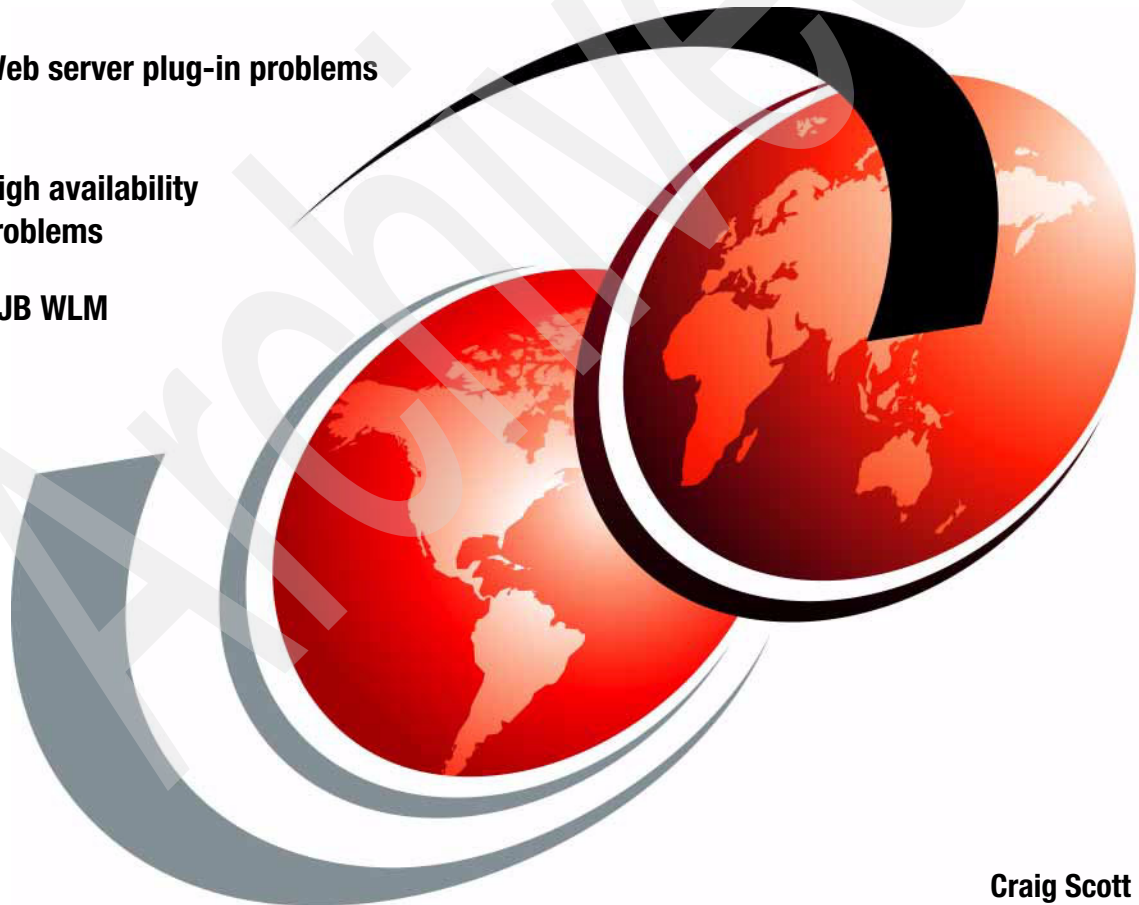


WebSphere Application Server V6.1: Workload Management Problem Determination

Diagnose Web server plug-in problems

Diagnose high availability
manager problems

Diagnose EJB WLM
problems



Craig Scott



International Technical Support Organization

**WebSphere Application Server V6.1: Workload
Management Problem Determination**

July 2007

Archived

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

First Edition (July 2007)

This edition applies to WebSphere Application Server V6.1 for distributed and i5/OS platforms.

© Copyright International Business Machines Corporation 2007. All rights reserved.

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

Contents

Notices	vii
Trademarks	viii
Preface	ix
The team that wrote this paper	ix
Become a published author	x
Comments welcome	x
Chapter 1. WebSphere Application Server load balancing problem determination	1
1.1 Introduction to workload management	2
1.1.1 WebSphere workload management	2
1.1.2 High availability	3
1.2 Identify symptoms of a workload management problem	4
1.3 Determine problem type	5
Chapter 2. Web server plug-in load balancing problem determination ...	7
2.1 Check system integrity	8
2.1.1 Identify symptoms	9
2.2 Plug-in not load balancing as expected	10
2.2.1 Collect diagnostics	10
2.2.2 Analyze diagnostics	11
2.2.3 Root causes	17
2.2.4 Max connections for the server has been reached	18
2.2.5 Server weights not equal	19
2.2.6 Session affinity is skewing load distribution	19
2.2.7 Server being marked down unexpectedly	20
2.2.8 Validate the solution	20
2.3 Plug-in not detecting a failed server	20
2.3.1 Collect diagnostics	21
2.3.2 Analyze diagnostics	21
2.3.3 Root causes	23
2.3.4 Host down	23
2.3.5 Hung application server	23
2.3.6 Validate the solution	24
2.4 Plug-in not recovering from a server outage	24
2.4.1 Collect diagnostics	24
2.4.2 Analyze diagnostics	25
2.4.3 Root causes	27

2.4.4	Tune the retry interval	27
2.4.5	Validate the solution	28
2.5	Session affinity not working	28
2.5.1	Collect diagnostics	28
2.5.2	Analyze diagnostics	28
2.5.3	Root causes	30
2.5.4	Session ID missing	30
2.5.5	Invalid clone ID	31
2.5.6	Validate the solution	32
2.6	Sessions not failing over under error conditions	32
2.6.1	Collect diagnostics	33
2.6.2	Analyze diagnostics	33
2.6.3	Root causes	36
2.6.4	Session replication incorrectly configured	36
2.6.5	Validate the solution	38
2.7	The next step	38
2.7.1	Java heap problems	38
2.7.2	Search online support	38
2.7.3	Re-evaluate the symptoms	39
2.7.4	Contact IBM	39
Chapter 3.	EJB workload management problem determination	41
3.1	Check system integrity	42
3.2	Identify symptoms	42
3.3	EJB application requests do not get serviced	43
3.3.1	Collect diagnostics	43
3.3.2	Analyze diagnostics	44
3.3.3	Root causes	47
3.3.4	No cluster data	47
3.3.5	CORBA errors	48
3.3.6	Validate the solution	50
3.4	EJB requests are not distributed evenly or to all servers	50
3.4.1	Collect diagnostics	50
3.4.2	Analyze diagnostics	50
3.4.3	Root causes	53
3.4.4	Unbalanced cluster member weights	53
3.4.5	Transaction affinity is skewing distribution	55
3.4.6	HA Manager is disabled	56
3.4.7	EJB server not in core group view	56
3.4.8	Validate the solution	57
3.5	Failing server still receives EJB requests (failover fails)	57
3.5.1	Collect diagnostics	57
3.5.2	Analyze diagnostics	57

3.5.3	Root causes	58
3.5.4	In-flight transaction trying to complete	58
3.5.5	Validate the solution	58
3.6	Restarted servers do not share the workload	59
3.6.1	Collect diagnostics	59
3.6.2	Analyze diagnostics	59
3.6.3	Root causes	60
3.6.4	Unusable interval has not expired	60
3.6.5	Validate the solution	60
3.7	The next step	61
Chapter 4. High availability manager problem determination		63
4.1	Determine if you need to use the HA Manager	64
4.1.1	Check system integrity	65
4.1.2	Identify symptoms	65
4.2	Singleton services not starting on failover	66
4.2.1	Collect diagnostics	66
4.2.2	Analyze diagnostics	66
4.2.3	Root causes	67
4.2.4	Inconsistent group membership	67
4.2.5	Thread pool problem	73
4.2.6	Network issues causing split views	73
4.2.7	Excessive core group sizes	75
4.3	Singleton services starting unexpectedly	76
4.4	Server will not start	76
4.4.1	Collect diagnostics	76
4.4.2	Analyze diagnostics	76
4.4.3	Root causes	77
4.4.4	Unable to obtain lock on transaction log	77
4.4.5	Validate the solution	77
4.5	Excessive resource usage	78
4.5.1	Collect diagnostics	78
4.5.2	Analyze diagnostics	78
4.5.3	Root causes	80
4.5.4	High JVM heap usage	80
4.5.5	Too much load due to HA Manager	81
4.5.6	Validate the solution	82
4.6	HA Manager messages in the logs	83
4.6.1	Collect diagnostics	83
4.6.2	Analyze diagnostics	83
4.7	The next step	84
4.7.1	Search online support	84
4.7.2	Re-evaluate the symptoms	84

Chapter 5. Collecting diagnostic data	87
5.1 Collecting JVM logs	88
5.2 Enabling the WLM trace	88
5.2.1 Enabling and gathering trace from a thin application client.	88
5.2.2 Enabling the trace from a J2EE application using launchClient . . .	89
5.2.3 Enabling the trace from the administrative console	90
5.3 Collecting the plug-in log	91
5.3.1 Setting the log level	91
Related publications	93
IBM Redbooks	93
Online resources	93
How to get Redbooks	95
Help from IBM	96
Index	97

Notices

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

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

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

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

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

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

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

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

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


COPYRIGHT LICENSE:

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

Trademarks

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

i5/OS®
DB2®
IBM®

Rational®
Redbooks®
Redbooks (logo) ®

Tivoli®
WebSphere®

The following terms are trademarks of other companies:

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

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

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

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

Preface

This IBM® Redpaper helps you to debug common problems that are related to workload management in WebSphere® Application Server network deployment on distributed and i5/OS® platforms. It discusses the following areas:

- ▶ High availability manager
- ▶ EJB™ workload management
- ▶ Web server plug-in load balancing

The team that wrote this paper

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Craig Scott is a Software Support Specialist in Australia. He has 18 years of experience in IT and over seven years experience in WebSphere Application Server. For the past three years, he has been working for IBM Software Support Center, assisting people to resolve problems with WebSphere Application Server from Version 4 through Version 6.1. He holds a degree in Computer Science from the University of Canberra. His areas of expertise include WebSphere Application Server, WebSphere Edge Server, IBM DB2®, and IBM Content Manager.

Thanks to the following people for their contributions to this project:

Carla Sadtler
International Technical Support Organization, Raleigh Center

Kevin Grigorenko
IBM WebSphere Serviceability team

Andrew Lam
IBM WebSphere Serviceability team

Mahesh Rathi
IBM WebSphere Serviceability team

Become a published author

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

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

- ▶ Use the online **Contact us** review Redbooks form found at:
ibm.com/redbooks
- ▶ Send your comments in an e-mail to:
redbooks@us.ibm.com
- ▶ Mail your comments to:
IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

WebSphere Application Server load balancing problem determination

Load balancing problems with WebSphere Application Server can occur at various stages of the process.

This paper covers the following:

- ▶ Workload management problems with the Web server plug-in
- ▶ EJB workload management problems
- ▶ High availability manager problems

This chapter is applicable to load balancing problems that occur on WebSphere Application Server V6.1 on distributed and i5/OS platforms.

1.1 Introduction to workload management

Workload management (WLM) is a WebSphere facility that provides load balancing and affinity between application servers in a WebSphere clustered environment. WLM is an important facet of performance. WebSphere uses workload management to send requests to alternate members of the cluster. WebSphere can also be configured to route concurrent requests from a user to the application server that serviced the first request. This is called session affinity and can be used to maintain a user's session over concurrent HTTP requests.

WLM is configurable. The administrator should ensure that each machine or server in the configuration processes a fair share of the overall client load that is being processed by the system as a whole. Workload should be spread among machines such that the workload corresponds to the machine processing power.

1.1.1 WebSphere workload management

Clustering application servers that host Web containers automatically enables plug-in workload management for the application servers and the servlets they host. Routing of servlet requests occurs between the Web server plug-in and the clustered application servers using HTTP or HTTPS as shown in Figure 1-1.

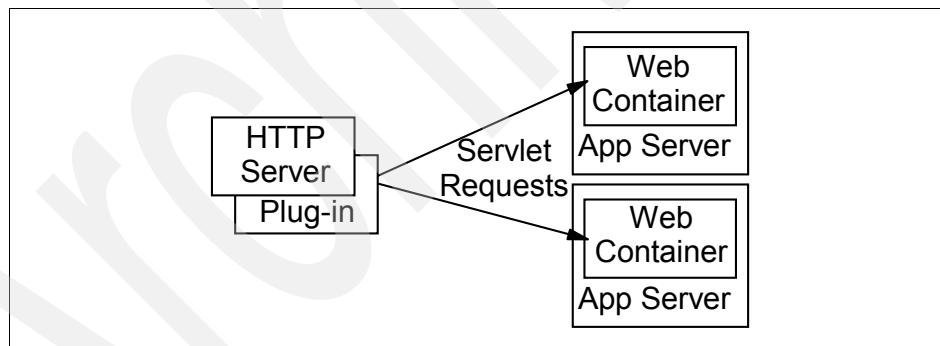


Figure 1-1 Plug-in (Web container) workload management

This routing is based on weights that are associated with the cluster members. If all cluster members have identical weights, the plug-in sends an equal number of requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range from zero to 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of zero. Weights can be changed dynamically during runtime by the administrator.

The Web server plug-in temporarily routes around unavailable cluster members.

Multiple application servers with the EJB containers can be clustered, enabling the distribution of EJB requests between the EJB containers as shown in Figure 1-2.

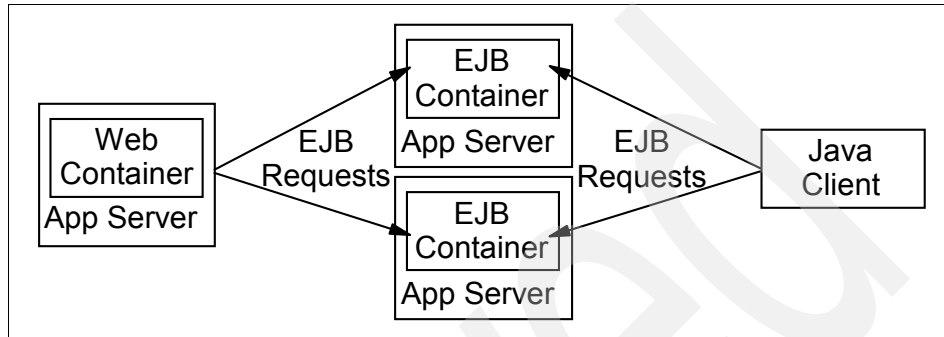


Figure 1-2 EJB workload management

In this configuration, EJB client requests are routed to available EJB containers in a round-robin fashion based on assigned server weights. The EJB clients can be servlets operating within a Web container, stand-alone Java™ programs using RMI/IIOP, or other EJBs.

The server-weighted round-robin routing policy ensures a distribution based on the set of server weights that have been assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers. The policy ensures the desired distribution, based on the weights that are assigned to the cluster members.

You can also choose to have requests sent to the node on which the client resides as the preferred routing. In this case, only cluster members on that node are chosen (using the round-robin weight method). Cluster members on remote nodes are chosen only if a local server is not available.

1.1.2 High availability

High availability is a concept closely related to workload management. Using high availability features is designed to eliminate single point of failures in the system. When one machine or server becomes unavailable, the workload is shifted to another.

Because problems during failover can affect workload balancing and performance, this paper includes problem determination guidance for the high availability manager.

1.2 Identify symptoms of a workload management problem

The primary symptom of a workload management problem is degradation of performance or problems during failover.

Specific symptoms of a problem with the Web server plug-in include:

- ▶ Plug-in not load balancing as expected
- ▶ Plug-in not detecting a failed server
- ▶ Plug-in not recovering from a server outage
- ▶ Session affinity not working
- ▶ Sessions not failing over under error conditions

Specific symptoms of a problem with EJB workload management include:

- ▶ EJB application requests do not get serviced
- ▶ EJB requests are not distributed evenly or to all servers
- ▶ Failing server still receives EJB requests (failover fails)
- ▶ Restarted servers do not share the workload

The following are typical symptoms of an HA Manager problem.

- ▶ Singleton services not starting on failover
- ▶ Singleton services starting unexpectedly
- ▶ Server will not start
- ▶ Excessive resource usage
- ▶ HA Manager messages in the logs. These will be prefixed with HMGR, CWRLS or DCSV.

1.3 Determine problem type

Workload management problems can occur at various stages depending on your configuration. The following is an approach you can take to navigate through this paper and diagnose a workload management problem.

1. Are you using clustering?
 - a. If not, then the activities in this Redpaper will not help you.
 - b. If yes, go to step 2.
2. Start the problem determination process by verifying that the Web server plug-in is working correctly and spreading work across the application servers as intended. Chapter 2, “Web server plug-in load balancing problem determination” on page 7 can guide you through this process.

If you determine that there is no problem with the Web server plug-in, continue to step 3.

3. Continue the process by verifying that requests for EJBs are being distributed among the application servers in the cluster as you intended. Chapter 3, “EJB workload management problem determination” on page 41 can guide you through this process.

If you determine there is no problem with the EJB workload management continue to step 4.

4. Are you using the high availability manager? This is the default. Chapter 4, “High availability manager problem determination” on page 63 can guide you through the process of determining whether you have a problem in the High Availability Manager.
5. If, after going through this process, you still have an undiagnosed problem, we recommend that you go back to *Approach to Problem Determination in WebSphere Application Server V6* at:

<http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf>

Review the problem classifications to see if there are any other components that might be causing the problem.

Web server plug-in load balancing problem determination

The Web server plug-in handles communications between a Web server and the WebSphere Application Server. The plug-in can direct requests to multiple WebSphere Application Servers to provide load balancing and failover functionality. Problems with the plug-in can cause performance to degrade or processing to behave abnormally.


The plug-in provides load balancing to direct requests to each server in a WebSphere Application Server cluster in turn. It chooses which server to send a request to, based on either a weighted round robin algorithm or by random distribution. The plug-in also supports session affinity where an HTTP request is routed back to the same application server that originally created the HTTP session for that user.

The plug-in also provides failover capability, where one or more of the servers that it is directing requests to is shut down or “crashes”. The plug-in will detect that an application server is no longer responding to requests and will stop sending requests to that server. The plug-in will periodically check back with the server to see if it has become available again.

2.1 Check system integrity

If you think you might have a Web server plug-in WLM problem, the first step is to verify the system integrity:

- Verify that all cluster member servers are up and running. In the administrative console, navigate to **Servers** → **Clusters** and ensure your clusters are showing as started as in Figure 2-1. All cluster members need to be started to ensure even load balancing.



New Delete Start Stop Ripplestart ImmediateStop		
Select	Name	Status
<input type="checkbox"/>	eibcluster1	
<input type="checkbox"/>	webcluster1	

Figure 2-1 Started clusters

- Ensure all cluster members are responding as expected. You can do this by using your Web browser to connect to each application server directly rather than through the HTTP server and plug-in.

First determine the HTTP transport port number (or HTTPS transport port number if using SSL) and then access the application URL using the application server host name and port number from your Web browser

You can determine the HTTP and HTTPS transport port number for each application server from a text file that is in each profile's log directory named `AboutThisProfile.txt`. This file will list the HTTP transport port number as shown in Example 2-1:

Example 2-1 `AboutThisProfile.txt`

```
Application server environment to create: Application server
Location: /opt/IBM/WebSphere/AppServer/profiles/Node2
Disk space required: 200 MB
Profile name: Node1
Make this profile the default: False
Node name: IBM99TVXRNode1
Host name: IBM99TVXRD.au.ibm.com
Enable administrative security (recommended): False
Administrative console port: 9061
Administrative console secure port: 9044
```

HTTP transport port: 9081
HTTPS transport port: 9444
Bootstrap port: 2810
...

For example: Assume you normally access your application via an external URL such as `www.ibm.com`. This URL points to a Web server with a WebSphere plug-in that load balances requests across the two servers named `appserver1` and `appserver2`. You would normally connect to your application using a URL like the following:

`http://www.ibm.com/wlm/BeenThere`

After determining the HTTP transport port number of the application server running on `appserver2` to be 9081, you would access the application from your browser via:

`http://appserver2:9081/wlm/BeenThere`

► Check your plug-in fix pack level

The plug-in fix pack level must be equal to or higher than all of the application servers that it is routing requests to. If you have applied fix packs to your application servers, ensure you have also applied the equivalent fix packs to your plug-in. You can check the fixpack levels of WebSphere Application Server and the WebSphere plug-in using the `versionInfo.bat` or `versionInfo.sh` script that is provided in the `bin` directory of each installation.

See the following URL for more information about supported combinations of Web server plug-in and WebSphere Application Server:

- Web server plug-in policy for WebSphere Application Server

<http://www-1.ibm.com/support/docview.wss?uid=swg21160581>

Tip: Your deployment manager must also be at a higher fix pack level than the application servers it manages. It is simplest to keep all components at the same fix pack level.

2.1.1 Identify symptoms

The following are typical symptoms of a plug-in WLM problem.

- Plug-in not load balancing as expected
- Plug-in not detecting a failed server
- Plug-in not recovering from a server outage

- ▶ Session affinity not working
- ▶ Sessions not failing over under error conditions

2.2 Plug-in not load balancing as expected

This section discusses steps to diagnose your problem if you have noticed that application requests are not being distributed equally to all application servers in the cluster

2.2.1 Collect diagnostics

The most important source of diagnostic data for analyzing plug-in load balancing problems is the plug-in log.

You should also review PMI data from each application server to monitor the number of requests being served and also server host metrics to ensure servers are not being overloaded.

You may also need to review logs from all the application servers in the cluster to determine if a particular cluster member is experiencing some problem that prevents it from servicing requests.

Collect the following:

- ▶ Plug-in log
See 5.3, “Collecting the plug-in log” on page 91.
- ▶ Application server logs:
 - JVM™ SystemOut and SystemErr logs
See 5.1, “Collecting JVM logs” on page 88.
 - native_stderr.log
By default, this file is located in
profile_root/logs/server_name/native_stderr.log

Depending on what you find, you may also need to look at the following:

- ▶ PMI data
- ▶ Server performance metrics

2.2.2 Analyze diagnostics

You will first need to review the plug-in log to determine if a server has been marked down.

Analyze the plug-in log

Look for the following in http-plugin.log

- Messages to indicate that a server has been marked down and that the plug-in is no longer sending it requests. Example 2-2 shows an application server being marked down as the plug-in is no longer able to connect to server.

Example 2-2 Server marked down

```
[Thu Apr 12 15:20:13 2007] 00001288 00001f90 - ERROR: ws_common:
websphereGetStream: Failed to connect to app server on host
'IBM99TVXRD.au.ibm.com', OS err=10061
[Thu Apr 12 15:20:13 2007] 00001288 00001f90 - ERROR: ws_common:
websphereExecute: Failed to create the stream
[Thu Apr 12 15:20:13 2007] 00001288 00001f90 - ERROR: ws_server:
serverSetFailoverStatus: Marking IBM99TVXRDNode1_server1 down
[Thu Apr 12 15:20:13 2007] 00001288 00001f90 - ERROR: ws_common:
websphereHandleRequest: Failed to execute the transaction to
'IBM99TVXRDNode1_server1' on host 'IBM99TVXRD.au.ibm.com'; will try another one
```

If you find a server is marked down, restart the server and then go to “Validate the solution” on page 20.

If you find an application server is being marked down, but you determine the server is not actually down, go to “Server being marked down unexpectedly” on page 20.

If this is not the cause of the problem, then you will need to determine the load balance distribution to investigate further by enabling plug-in detail logging and by reviewing PMI data.

Analyze the plug-in detail logging

To verify plug-in load balancing, you will need to enable further logging at the plug-in. The default level of logging is Error. Increase your plug-in log level and then run a test to recreate the issue. If you have a simple environment with only one clustered application, you can use the Stats logging level. If your environment is complex, use the Trace logging level. See 5.3.1, “Setting the log level” on page 91 for more information on changing the plug-in log tracing level.

Look for the following:

- Using the Warn logging level, you may see the following message:

```
[Fri May 04 11:39:10 2007] 000000d8 000013e0 - WARNING:  
ws_server_group: serverGroupCheckServerStatus: Server  
IBM99TVXRNode2_server2 has reached maximum connections and is not  
selected
```

Go to “Max connections for the server has been reached” on page 18.

- ▶ Using the Stats logging level, you will see output as shown in Example 2-3. This is showing the distribution of requests across the application servers. However Stats does not show the different URI requests. When you have multiple applications in your environment, you will not be able to determine the distribution of requests for the different applications.

Example 2-3 Stats output

```
[Thu Apr 12 14:31:29 2007] 000018f4 00001b70 - STATS: ws_server:  
serverSetFailoverStatus: Server IBM99TVXRNode2_server2 : pendingRequests 0  
failedRequests 0 affinityRequests 0 totalRequests 54.  
[Thu Apr 12 14:31:33 2007] 000018f4 00001b70 - STATS: ws_server:  
serverSetFailoverStatus: Server IBM99TVXRNode1_server1 : pendingRequests 0  
failedRequests 0 affinityRequests 0 totalRequests 55.
```

- ▶ Using the **Trace** log level, you will get sufficient information to determine the distribution of requests based on the applications running in your environment. However, the amount of data generated will be large as it will log the sequence of events that occurs as each request that comes through the plug-in is processed. You should only run with this logging level when necessary.

In the Web server plug-in configuration file, you can see that the context root from each installed application is mapped to an application server or cluster that can handle that request.

To process the results from the http-plugin.log file, you will need to look through the file and extract the data that shows request distribution.

Example 2-4, shows extracts from a plug-in Trace of a request to the snoop servlet.

You can see the request come in to the plug-in, the plug-in tells you it is using a round robin algorithm and that it has chosen **IBM99TVXRNode2_server2** to process the request. Near the end of the request output, the plug-in prints the overall statistics for the chosen server.

Example 2-4 Extracting request distribution by URI

```
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DEBUG: lib_util:  
parseHostHeader: Defaulting port for scheme 'http'  
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DEBUG: lib_util:  
parseHostHeader: Host: 'ibm99tvxrd', port 80
```

```
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DEBUG: ws_common:
websphereCheckConfig: Current time is 1176352824, next stat time is
1176352859
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DETAIL: ws_common:
websphereShouldHandleRequest: trying to match a route for:
vhost='ibm99tvxrd'; uri='/snoop'
...
Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DEBUG: ws_server_group:
serverGroupNextRoundRobinServer: Round Robin load balancing
...
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - DEBUG: ws_server_group:
serverGroupNextRoundRobinServer: use server IBM99TVXRDNode2_server2
...
[Thu Apr 12 14:40:24 2007] 0000190c 000018e8 - STATS: ws_server:
serverSetFailoverStatus: Server IBM99TVXRDNode2_server2 :
pendingRequests 0 failedRequests 0 affinityRequests 0 totalRequests 85.
```

A single request to the snoop servlet at **Trace** log level will produce 187 lines of logging data in the http-plugin.log. In a production environment with high transaction rates, the amount of data to parse is going to get very large very quickly and it is not practical to process it by hand. For this reason you should consider using a scripting language such as Python or Perl to assist you process this data and generate summary reports.

If you confirm that you are experiencing uneven load balancing, then you will need to look further to determine why.

- ▶ Check the plug-in configuration to ensure the server weights are set as appropriate. For more information, go to “Server weights not equal” on page 19.
- ▶ If you are seeing a server not being selected, as the maximum number of requests has been reached, go to “Max connections for the server has been reached” on page 18.
- ▶ Plug-in logging levels Stats and higher will also report on requests that have been routed to maintain session affinity. If you are seeing uneven load balancing due to session affinity, go to “Session affinity is skewing load distribution” on page 19.

Review the application logs, PMI statistics and server metrics to determine why a server is not processing requests, or processing less requests than you expect.

Analyze the application server logs

For each application server that is not participating in the load balancing, look for exceptions or errors in the SystemOut or SystemErr log that might indicate why

the application server is not servicing requests. Some of the problems you might find include:

- ▶ The application server has crashed and is restarting
- ▶ The application has hung and is not processing requests
- ▶ The application server has run out of memory
- ▶ Other application related errors to indicate why requests cannot be processed. Note however that these will not typically cause the plug-in to mark the server down, as the server is still responding to requests, just with an error.

Note: The resolution of these issues is outside the scope of this paper.

- ▶ Check verbose GC data

If your JVM heap size usage is getting close to its maximum, you are likely to see an excessive number of garbage collections being performed. Frequent garbage collections in the JVM will increase the CPU being used on the server and reduce the number of requests that can be processed.

Verbose GC can be enabled dynamically on a running server through the administrative console, see Figure 2-2.

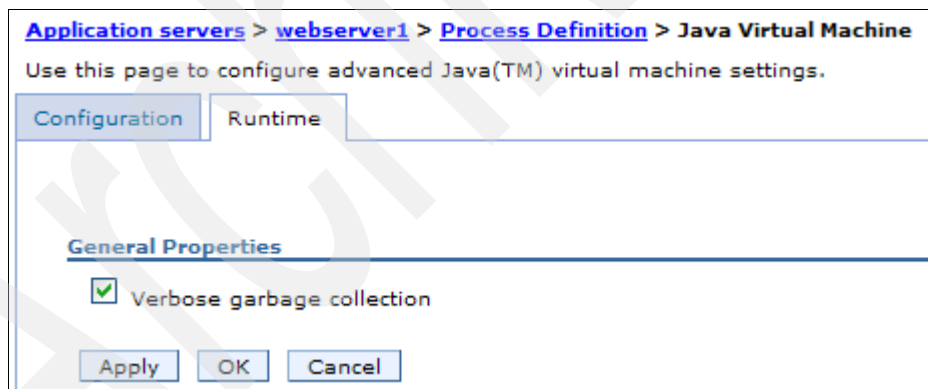


Figure 2-2 Enable verbose GC in the admin console

If you find you are experiencing excessive Java heap usage, you will need to pursue that as the root cause. Java heap problems are outside the scope of this paper. For a list of references that can help you pursue this type of problem, see 2.7.1, “Java heap problems” on page 38.

- Look for an out of memory condition

If your available JVM heap size or process native memory has been exhausted, you will see an out of memory exception reported in the logs.

In the case of JVM heap size exhaustion, you will see excessive garbage collection activity leading up to this point. When the memory is exhausted, the server will report the condition and generate a javacore and heapdump file for your analysis. The server may continue to function or it may crash depending on the circumstances. The JVM may continue to try and service requests.

In the case of native memory exhaustion, the JVM will crash and will generate a javacore, core file or user.dmp file. The JVM will no longer participate in servicing requests.

If you find you are experiencing excessive Java heap usage, you will need to pursue that as the root cause. Java heap problems are outside the scope of this paper. For a list of references that can help you pursue this type of problem, see 2.7.1, “Java heap problems” on page 38.

Performance Monitoring Infrastructure (PMI) data

Check the PMI data to ensure all cluster members are processing the expected number of requests and there are no adverse statistics such as high CPU or JVM memory usage that might indicate a problem.

Look for the following:

- You can review the number of requests being processed by each application server using the Tivoli® Performance Viewer (TPV) or other PMI client. TPV will also show other problems such as excessive heap usage or garbage collection.

Navigate to **Monitoring and tuning** → **Performance viewer** → **Current activity** and then click each server name in the cluster in turn. Figure 2-3 shows the initial view you will see when opening the viewer and will show you the number of requests serviced by each servlet or JavaServer Pages (JSP™) and the average response time. Comparing these statistics across servers will show the distribution of load balanced requests from the application server point of view. Compare this to what you expect to see from the statistics gathered from the plug-in.

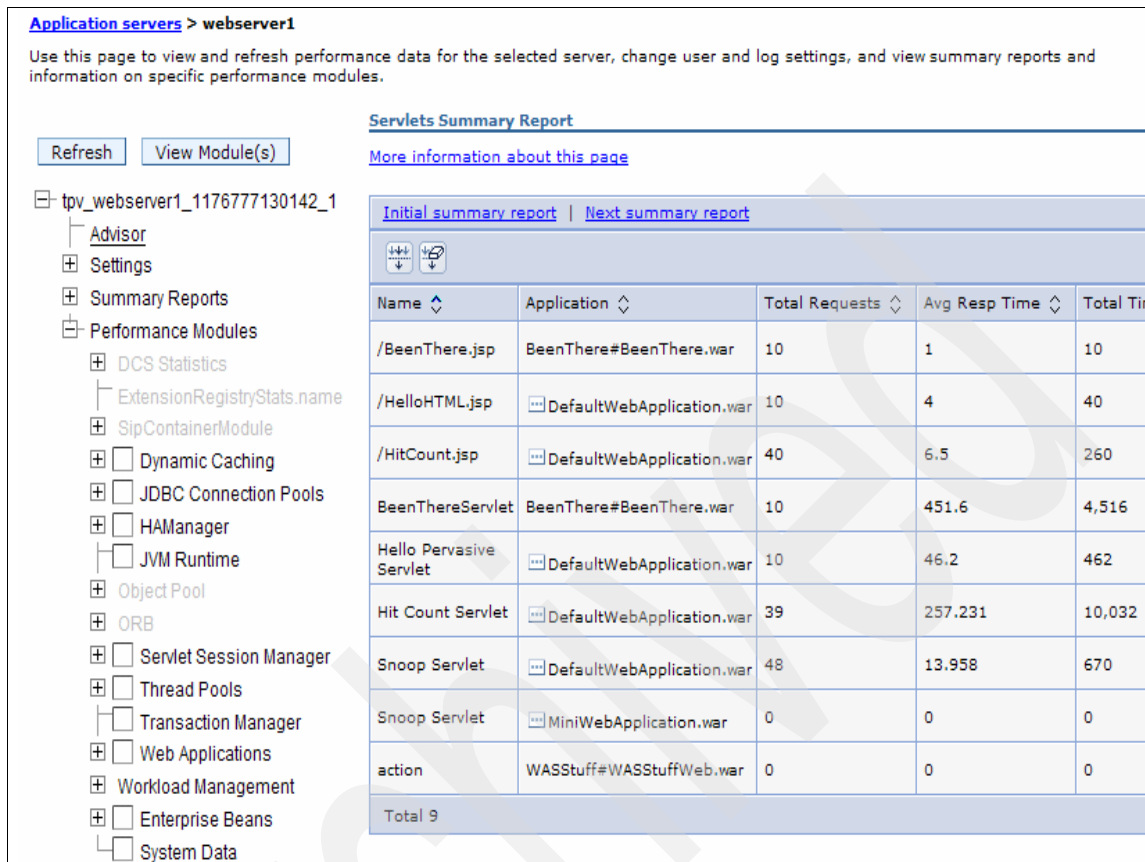


Figure 2-3 PMI Statistics

If you find uneven processing of requests but there are no messages in the plug-in log to indicate why, you will need to increase the plug-in logging level to further debug this issue. Go to “Analyze the plug-in detail logging” on page 11.

- Review the JVM heap usage in the TPV to get an indication of the health of the application server. Navigate to **Monitoring and tuning** → **Performance viewer** → **Current activity** and then click each server name in the cluster in turn. Expand the **Performance modules** tree in the left hand side of the TPV and click **JVM Runtime**. Then click **Show modules**. Figure 2-4 is an example of a healthy JVM heap usage.

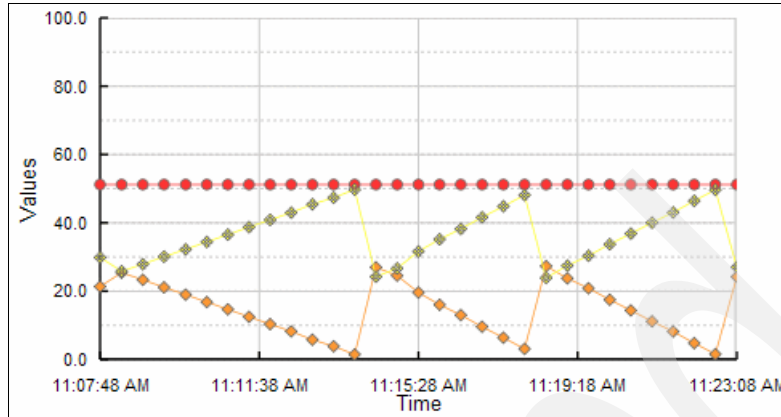


Figure 2-4 JVM heap usage

If you find you are experiencing excessive Java heap usage, you will need to pursue that as the root cause. Java heap problems are outside the scope of this paper. For a list of references that can help you pursue this type of problem, see 2.7.1, “Java heap problems” on page 38.

Server metrics

Server metrics such as CPU, I/O and page space utilization can also show uneven or unexpected load balancing. You should use the tool most appropriate to your operating system to check these statistics. For example, in Windows® you could use **Performance Monitor**. In a Unix or Linux® environment, you could use **vmstat** or **top**.

Look for the following:

- ▶ Unexpectedly high CPU, I/O or page space usage on one server compared to the others. This could be indicative of uneven load balancing. However it could also be indicative of other problems such as a non WebSphere related process using too much CPU, excessive garbage collection or a looping WebSphere Application Server.

Resolution of these issues is outside the scope of this paper.

2.2.3 Root causes

The following are possible root causes for uneven load balancing:

- ▶ Max connections for the server has been reached
- ▶ Server weights not equal
- ▶ Session affinity is skewing load distribution

- Server being marked down unexpectedly

2.2.4 Max connections for the server has been reached

If your plug-in has been configured to restrict the number of connections that a server can accept (maxConnections), you may be reaching this maximum. Example 2-5 shows a server that has reached this limit with the logging level set to Stats.

Example 2-5 Max connections reached

```
[Fri Apr 20 15:49:11 2007] 000008d8 000000d0 - STATS: ws_server_group:  
serverGroupCheckServerStatus: Checking status of IBM99TVXRNode2_server2,  
ignoreWeights 0, markedDown 0, retryNow 0, retryInSec --, wlbAllows 11  
reachedMaxConnectionsLimit 1
```

Resolve the problem

maxConnections is configured individually for each server in a cluster by editing the plug-in configuration file. You can determine the path to the configuration file through the WebSphere admin console. Navigate to **Web servers** → **web_server_name** → **Plug-in properties** and look for Web server copy of Web server plug-in files. You will find the path to the running copy of the Web server plug-in configuration file in the Plug-in configuration directory and file name box, for example:

```
/opt/IBM/WebSphere/IHS/Plugins/config/webserver/plugin-cfg.xml
```

Example 2-6 shows the default setting of the MaxConnections parameter on server2 from the plug-in configuration file.

Example 2-6 Max connections configuration

```
<Server CloneID="1251slkq" ConnectTimeout="0" ExtendedHandshake="false"  
LoadBalanceWeight="2" MaxConnections="-1" Name="IBM99TVXRNode2_server2"  
ServerIOTimeout="0" WaitForContinue="false">
```

The default value of MaxConnections is “-1” which means no limit is imposed by the plug-in. If you find MaxConnections has been set to a specific value, you will need to determine why it was set from the system administrator who made this change. The value chosen may no longer be valid and should be reviewed.

Note: It is not possible to modify MaxConnections through the WebSphere administrative console.

2.2.5 Server weights not equal

In the configuration of the plug-in, you can set server weights that will adjust the ratio of requests that are directed to each server. For example, if you have two servers where one server is four times more powerful than the other, you might adjust server weights so the more powerful server services four requests to every one serviced by the other.

If you do this, you will not see even distribution of requests among your application servers by design. A more detailed discussion of this can be found at:

- Understanding IBM HTTP Server plug-in Load Balancing in a clustered environment

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219567>

Resolve the problem

Adjust server weights as appropriate for each of the cluster member servers.

2.2.6 Session affinity is skewing load distribution

Session affinity can cause the plug-in to distribute requests unevenly.

Example 2-7 shows an imbalance in load balancing caused by session affinity. You can see that server1 has serviced 448 requests while server2 has only serviced 45 requests, a ratio of roughly 10 to 1. However you can also see that of the 448 requests serviced by server1, 405 of these were sent to that server to maintain session affinity. Without the session affinity requests, the load balance ratio is almost 1 to 1.

Example 2-7 http-plugin.log showing skewed load balancing due to session affinity

```
[Thu Apr 12 16:26:15 2007] 00001288 00001f90 - STATS: ws_server:
serverSetFailoverStatus: Server IBM99TVXRNode1_server1 : pendingRequests 0
failedRequests 0 affinityRequests 405 totalRequests 448.
[Thu Apr 12 16:26:16 2007] 00001288 00001300 - STATS: ws_server:
serverSetFailoverStatus: Server IBM99TVXRNode2_server2 : pendingRequests 0
failedRequests 0 affinityRequests 0 totalRequests 45.
```

Resolve the problem

Adjust server weights as per the following technote to try and balance server affinity based requests more appropriately.

- Understanding IBM HTTP Server plug-in Load Balancing in a clustered environment

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219567>

This situation is often seen in test environments and this imbalance will probably sort itself out as the number of distinct concurrent users with unique sessions increases. As the number of unique sessions increases, the plug-in will be able to distribute these more evenly across the cluster members.

2.2.7 Server being marked down unexpectedly

In some instances, you may find that the plug-in is marking an application server down for no apparent reason. You can access the application on the server's HTTP transport port yet the plug-in log is reporting that the server has been marked down.

Example 2-2 on page 11 shows the error you will see in the plug-in log.

Resolve the problem

This could be caused by intermittent error conditions on the application server. If the application server is unable to process a connection for a particular request, the plug-in will mark the server down. This could be due to resource constraints such as insufficient memory to establish a TCP/IP connection or some other error condition. Check the application server logs.

You can also see this condition intermittently if the plug-in fix pack level is lower than the application server fix pack level. Apply the appropriate fix pack to your plug-ins.

2.2.8 Validate the solution

Recreate the situation and validate that requests are being distributed to the clustered servers as you expect.

If this does not resolve the issue, go to “The next step” on page 38.

2.3 Plug-in not detecting a failed server

This section discusses steps to diagnose the problem where an application server is no longer responding to requests, yet the plug-in is still attempting to

send requests to that server. This situation will cause a delay in processing while the plug-in waits for the server to respond or for the TCP connection to time-out.

2.3.1 Collect diagnostics

You will need to review a plug-in trace to see why the plug-in is trying to send requests to a server that is not responding. You will also need to review the JVM logs from the application server that is failing or has failed.

- ▶ Plug-in log using the Trace logging level.
See 5.3, “Collecting the plug-in log” on page 91.
- ▶ JVM SystemOut and SystemErr logs
See 5.1, “Collecting JVM logs” on page 88.

2.3.2 Analyze diagnostics

You will first need to set the plug-in logging level to Trace, recreate the problem and review the trace data generated in the `http-plugin.log`

Analyze the plug-in trace

Look for the following:

- ▶ The plug-in establishing a connection to an application server and not getting a response.

In Example 2-8, you can see the plug-in choosing a server, `server1`, and building the request stream to be passed to the application server. It creates the stream to the application server and sends the request.

Example 2-8 A request with no response

```
[Fri May 04 12:34:27 2007] 00001660 000019d4 - TRACE: ws_server_group:
serverGroupIncrementConnectionCount: Server IBM99TVXRDNode1_server1 picked,
pendingConnectionCount 2 totalConnectionsCount 2.
[Fri May 04 12:34:27 2007] 00001660 000019d4 - DEBUG: ws_server_group:
serverGroupNextRoundRobinServer: use server IBM99TVXRDNode1_server1
[Fri May 04 12:34:27 2007] 00001660 000019d4 - TRACE: ws_common:
websphereFindTransport: Finding the transport
[Fri May 04 12:34:27 2007] 00001660 000019d4 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): IBM99TVXRD.au.ibm.com on
port 9081
[Fri May 04 12:34:27 2007] 00001660 000019d4 - TRACE: ws_common:
websphereExecute: Executing the transaction with the app server
...
```

```
[Fri May 04 12:34:27 2007] 00001660 000019d4 - TRACE: ws_common:
websphereExecute: Wrote the request; reading the response
[Fri May 04 12:34:27 2007] 00001660 000019d4 - DETAIL: lib_htresponse:
htresponseRead: Reading the response: 4b1b18c
[Fri May 04 12:34:27 2007] 00001660 00001388 - DEBUG: ws_common:
websphereGetStream: socket 260 connected to IBM99TVXRD.au.ibm.com:9081
[Fri May 04 12:34:27 2007] 00001660 00001388 - DEBUG: lib_stream: openStream:
Opening the stream
```

The last entry for the thread ID (000019d4) is:

```
Fri May 04 12:34:27 2007] 00001660 000019d4 - DETAIL:
lib_htresponse: htresponseRead: Reading the response: 4b1b18c
```

The plug-in log then shows processing for a new request as shown by the change in the thread ID to 00001388. Searching down through the log will not show a response for that particular request and thread ID.

You will need to look into the JVM logs to see why the application server is not responding.

Analyze the JVM logs

Look for the following:

- ▶ If you are unable to attach to the host to view the logs, the server has crashed.

Resolution of this problem is outside the scope of this chapter, but you can tune the plug-in to better recognize this situation. Go to “Host down” on page 23.

- ▶ WSVR0605W messages indicating hung threads.

The following message indicates your application server has hung or is in the process of hanging:

```
[1/05/07 15:19:57:873 EST] 0000001f ThreadMonitor W WSVR0605W:
Thread "WebContainer : 1" (0000003f) has been active for 695740
milliseconds and may be hung. There is/are 99 thread(s) in total in
the server that may be hung.
```

You might also see the process still running but no further logging activity.

Resolution of this problem is outside the scope of this chapter, but you can tune the plug-in to better recognize this situation. Go to “Hung application server” on page 23.

2.3.3 Root causes

The following are possible root causes that would lead to these issues. The resolution of these root causes is outside the scope of this chapter. However you can tune the plug-in to better detect and work around the problem.

- ▶ Host down
- ▶ Hung application server

2.3.4 Host down

If the server host is completely down, the plug-in can take some time to recognize this. In this case, the plug-in will attempt to connect to the server but since it is no longer responding to TCP requests, the plug-in will need to wait until the operating system times out the connection request before marking the server down. Depending on the operating system, this can take up to 3 minutes.

Resolve the problem

The plug-in can be tuned to better recognize a server host that has gone down by adjusting the `ConnectTimeout` parameter.

Tuning the `ConnectTimeout` parameter is discussed in the technote:

- ▶ Understanding HTTP plug-in failover in a clustered environment
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219808>

2.3.5 Hung application server

If an application server has hung, it can still be possible for it to accept incoming HTTP requests yet do no work with them. In this case, the plug-in will be able to establish a connection with the application server and so will not recognize that it has failed. It will send the request across and wait on a response. In this instance, attempting to connect to the applications server's HTTP transport port will cause your browser to sit and wait for a response.

Resolve the problem

The plug-in can be tuned to mark a hung server down. Adjust the `ServerIOTimeout` parameters to tune the plug-in so that it recognizes a hung server as an error condition.

ServerIOTimeout is discussed in the technote:

- Understanding HTTP plug-in failover in a clustered environment

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219808>

2.3.6 Validate the solution

After making the changes described here, recreate the issue and ensure that the plug-in marks the server down in a timely manner. This will enable the cluster members to carry on processing while you resolve the application server problems.

If you are still experiencing an issue with the plug-in not marking the server down, go to “The next step” on page 38.

2.4 Plug-in not recovering from a server outage

This section discusses steps to diagnose the problem where following a server failure, the plug-in marked the server down and stopped routing requests to that server. You have restored the application server but the plug-in has not started sending it requests.

2.4.1 Collect diagnostics

You will need to collect and review a plug-in trace to see why the plug-in is not sending requests to the restored server. You may also need to review the JVM logs from the application server.

Depending on the plug-in logging level you have set, you may need to set it higher and recreate the problem to collect the data.

Collect the following:

- The plug-in log where the logging level has been set to a higher level than Error.

See 5.3, “Collecting the plug-in log” on page 91.

- JVM SystemOut and SystemErr logs

See 5.1, “Collecting JVM logs” on page 88.

2.4.2 Analyze diagnostics

You will first need to review the plug-in log. Depending on what you see, you may also need to review the JVM logs.

Analyze the plug-in trace

Look for the following in the trace:

- ▶ Setting the plug-in logging level to **Stats** will allow you to see when an application server starts responding to requests, you will first see the server being marked down. While it is down, there will be no stats lines written to the `http-plugin.log` file for that server. When the plug-in starts routing requests to the server, you will again see stats lines for the server.
- ▶ Setting the plug-in logging level to **Detail** or higher will show you more detailed information about the process the plug-in is taking to check a server that has been marked down as shown in Example 2-9. First, you can see `server1` being marked down. Processing the next request shows that the plug-in will retry `server1` in 5 seconds. Approximately 5 seconds later, the message reports that the server will be retried now. This retry fails. Finally, you can see the last request in the example being successfully routed to `server1`.

Example 2-9 Checking to see if a server is available

```
[Fri Apr 20 13:02:04 2007] 00001e44 00001058 - ERROR: ws_server:
serverSetFailoverStatus: Marking IBM99TVXRDNode1_server1 down
[Fri Apr 20 13:02:04 2007] 00001e44 00001058 - STATS: ws_server:
serverSetFailoverStatus: Server IBM99TVXRDNode1_server1 : pendingRequests 0
failedRequests 1 affinityRequests 0 totalRequests 4.
...
[Fri Apr 20 13:02:09 2007] 00001e44 00001058 - STATS: ws_server_group:
serverGroupCheckServerStatus: Checking status of IBM99TVXRDNode1_server1,
ignoreWeights 0, markedDown 1, retryNow 0, retryInSec 5, wlbAllows 0
reachedMaxConnectionsLimit 0
...
[Fri Apr 20 13:02:14 2007] 00001e44 00001058 - STATS: ws_server_group:
serverGroupCheckServerStatus: Checking status of IBM99TVXRDNode1_server1,
ignoreWeights 0, markedDown 1, retryNow 1, retryInSec 0, wlbAllows 0
reachedMaxConnectionsLimit 0
[Fri Apr 20 13:02:14 2007] 00001e44 00001058 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): IBM99TVXRD.au.ibm.com on
port 9081
[Fri Apr 20 13:02:14 2007] 00001e44 00001058 - ERROR: ws_common:
websphereGetStream: Failed to connect to app server on host
'IBM99TVXRD.au.ibm.com', OS err=10061
...
```

```
[Fri Apr 20 13:04:04 2007] 00001e44 00001058 - STATS: ws_server_group:
serverGroupCheckServerStatus: Checking status of IBM99TVXRDNode1_server1,
ignoreWeights 0, markedDown 1, retryNow 1, retryInSec 0, wlbAllows 0
reachedMaxConnectionsLimit 0
[Fri Apr 20 13:04:04 2007] 00001e44 00001058 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): IBM99TVXRD.au.ibm.com on
port 9081
[Fri Apr 20 13:04:04 2007] 00001e44 00001058 - DETAIL: ws_common:
websphereGetStream: Created a new stream; queue was empty, socket = 620
```

If you see the plug-in still not being able to connect to the restored server, you will need to review the JVM logs.

If you determine that the application server started, but the plug-in is taking too long to recognize this, go to “Tune the retry interval” on page 27.

Analyze the JVM logs

Look for the following:

- Ensure the server has finished starting by looking for the message that indicates a server is ready to process requests:

```
[3/05/07 17:01:21:959 EST] 0000000a WsServerImpl A WSVR0001I:
Server ejbserver1 open for e-business
```

If this message has not appeared, either wait for the server to finish startup or try restarting it again.

- Ensure the application has started correctly:

If the application fails to start, you will see exceptions that will describe what has gone wrong with the application on startup. Example 2-10 shows the snoop servlet failing to start due to a `ClassNotFoundException`.

Example 2-10 Application fails to start

```
[4/05/07 13:16:55:509 EST] 00000023 ServletWrapper E [Servlet
Error]-[SnoopServlet]: java.lang.ClassNotFoundException: SnoopServlet
    at
com.ibm.ws.classloader.CompoundClassLoader.findClass(CompoundClassLoader.java:4
72)
    at
com.ibm.ws.classloader.CompoundClassLoader.loadClass(CompoundClassLoader.java:3
73)
```

Resolution of this problem is outside the scope of this paper, but you can tune the plug-in timing of when to check if a server has become available.

2.4.3 Root causes

The root cause of this issue is not a problem with plug-in load balancing, however you can tune the plug-in to respond to a restored server in a more timely manner by tuning the retry interval.

2.4.4 Tune the retry interval

When an application server comes back online, the plug-in will automatically recognize this and start to route requests back to that server. The plug-in does not log a message to indicate a server is back online, it simply starts routing requests to that server.

The timing of how often the server checks to see if a server has become available is controlled by the **Retry interval** parameter set in the administrative console as shown in Figure 2-5. **Navigate to Servers → Web servers → server_name → Plug-in properties → Request routing.**

You can also see the selected value in the cluster definition in the plugin-cfg.xml file.

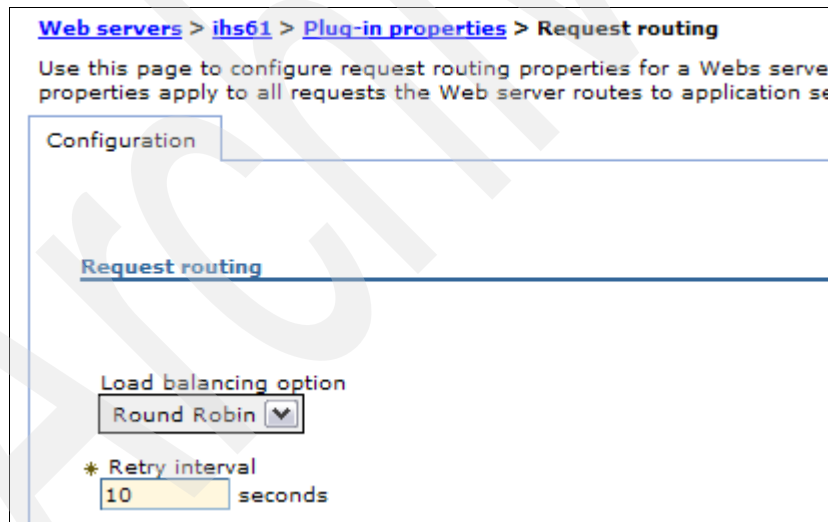


Figure 2-5 Retry interval

Resolve the problem

Tune the retry interval to suit your environment. If the servers typically take a minute to start, set the retry interval to say 30 seconds. If you set the interval too high, you might have an idle server while the plug-in waits for the retry interval to

elapse. If you set the retry interval too low, the plug-in will check too often to see if a server has become available.

2.4.5 Validate the solution

After tuning the plug-in retry interval, recreate the failure and restore the server. Ensure that the plug-in responds to a restored server in a timely manner. Also, ensure that you are not checking for a restored server too often as this could be cause a performance impact.

If you still see problems with the server not recognizing a restored server, go to “The next step” on page 38.

2.5 Session affinity not working

This section discusses steps to diagnose your problem if your application relies on session affinity to maintain user state information, but you are not seeing session affinity being honoured.

2.5.1 Collect diagnostics

You will need to collect and review a plug-in trace to see why the plug-in is not sending requests to the correct servers. You may also need to review the JVM logs from the application server.

Depending on the plug-in logging level you have set, you may need to set it higher and recreate the problem to collect the data.

Collect the following:

- ▶ The plug-in log where you have set the logging level to Trace.
See 5.3, “Collecting the plug-in log” on page 91.
- ▶ JVM SystemOut and SystemErr logs
See 5.1, “Collecting JVM logs” on page 88.

2.5.2 Analyze diagnostics

You will first need to review the plug-in log. Depending on what you see, you may also need to review the JVM logs.

Analyze the plug-in trace

Refer to “Analyze the plug-in detail logging” on page 11 for details on plug-in tracing.

Look for the following:

- Detailed information about the processing of session affinity requests.
Review the trace to determine why a particular request was sent to a particular server. Example 2-11 shows an example of a request to the HitCount servlet being routed to **server1** based on the partition ID located in the **JSESSIONID** cookie.

Example 2-11 Processing the HitCount servlet

```
[Tue Apr 17 16:49:57 2007] 00001b74 00000698 - DETAIL: ws_common:
websphereShouldHandleRequest: trying to match a route for: vhost='localhost';
uri='/hitcount'
...
[Tue Apr 17 16:49:57 2007] 00001b74 00000698 - DEBUG: ws_common:
websphereParseCloneID: Parsing clone ids from
'0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5'
[Tue Apr 17 16:49:57 2007] 00001b74 00000698 - DEBUG: ws_common:
websphereParseCloneID: Parsing clone ids from
'0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5'
[Tue Apr 17 16:49:57 2007] 00001b74 00000698 - DEBUG: ws_server_group:
serverGroupMatchPartitionID: Match found for partitionID '34J7RVRCR5'
[Tue Apr 17 16:49:57 2007] 00001b74 00000698 - DEBUG: ws_server_group:
serverGroupGetServerByID: Match for clone 'IBM99TVXRDNode1_server1'
```

The trace will also show you how the partition table is constructed as shown in Example 2-12. The partition table is populated when the first request is processed by the plug-in. It communicates with the application servers and builds the partition table based on the session failover configuration. The plug-in gets the partition table from the application servers as it needs to be able to maintain a consistent partition table to handle session failover among multiple HTTP servers or following an HTTP server restart.

Example 2-12 Plug-in building the partition table

```
[Tue Apr 17 16:57:30 2007] 00001584 000014b8 - TRACE: ws_common:
ParsePartitionIDs: Parsing partitionID pair from
'8I9FV6LVH:1251s1lkq;34J7RVRCR5:12522k3ef;'
[Tue Apr 17 16:57:30 2007] 00001584 000014b8 - TRACE: ws_common:
ParsePartitionIDs: Adding partitionID / clone pair '8I9FV6LVH' : '1251s1lkq'
[Tue Apr 17 16:57:30 2007] 00001584 000014b8 - TRACE: ws_common:
ParsePartitionIDs: Adding partitionID / clone pair '34J7RVRCR5' : '12522k3ef'
```

Being able to see how the partition table is setup will allow you to cross reference the partition ID to the clone ID of each application server. If you are not seeing a partition table, you should be seeing session affinity being maintained by the clone ID.

If you are not seeing the session ID being passed back from the client, for example, in the JSESSIONID cookie, go to “Session ID missing” on page 30.

If you are seeing that the session ID can not be matched to an existing clone ID or partition ID, go to “Invalid clone ID” on page 31.

If you are not seeing the cause of the problem in the plug-in trace, then you will need to review the JVM logs.

Analyze the JVM logs

Look for the following:

- ▶ Ensure that the application servers are processing the sessions correctly. You will need to take an HTTP session trace in WebSphere Application Server.

Debugging session handling problems in WebSphere Application Server applications is outside the scope of this document.

2.5.3 Root causes

The following are possible root causes for this issue:

- ▶ Session ID missing
- ▶ Invalid clone ID

2.5.4 Session ID missing

When an HTTP session is established, that session is assigned an ID and the ID is passed to the client. In subsequent requests, the session ID is passed back from the client so that the application can keep track of the session.

Resolve the problem

Ensure that the session ID can be passed back from the client using one of the available methods. WebSphere Application Server can maintain session information using any or all of the following:

- Cookies

The session ID is encoded in a session cookie called **JSESSIONID** by default. The client browsers must be set to accept cookies for this to work.

- URL rewriting

If cookies are not available, sessions can be maintained by rewriting the URL to include the session ID as an HTTP parameter. This introduces a performance impact.

- SSL ID tracking

The session identifier is linked to SSL identifier when SSL is used to secure browser to application server communications.

The allowed session tracking mechanisms are configured for the application servers in the administrative console. Navigate to **Servers** → **Application servers** → *server_name* → **Session management**. Figure 2-6 shows cookies set as the session tracking mechanism.

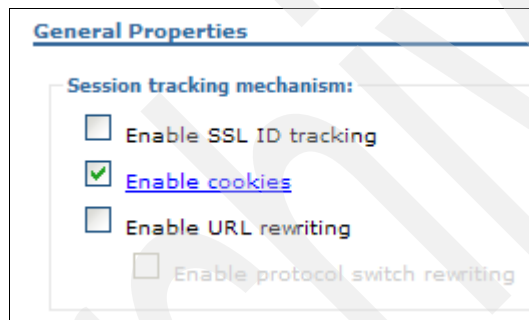


Figure 2-6 Session tracking mechanisms

2.5.5 Invalid clone ID

If the session ID being returned from the client contains a clone ID that does not match the clone IDs configured in the plug-in configuration file, then session affinity will not be maintained.

Resolve the problem

Ensure the session IDs and clone IDs being reported in the plug-in trace are valid. If the clone IDs have changed or the plug-in has been manually edited, then these values may no longer be valid. Regenerate and distribute the plug-in to ensure it reflects the latest configuration.

In the administrative console, navigate to **Servers** → **Web servers**. Select the Web server name as shown in Figure 2-7 and click **Generate Plug-in**. Click the server name again and click **Propagate plug-in**.

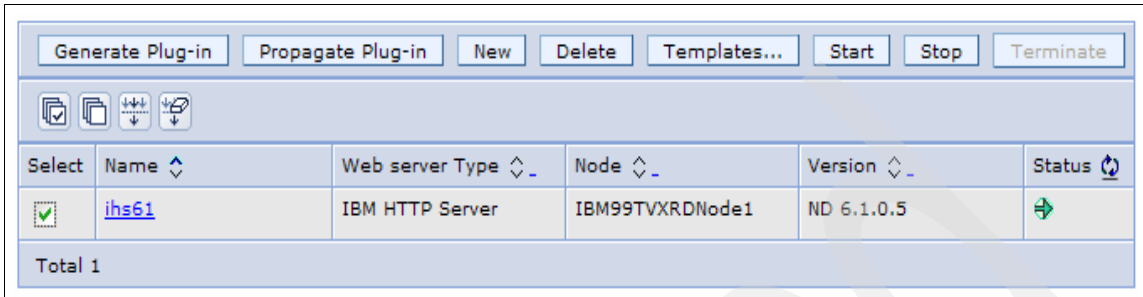


Figure 2-7 Generate the plug-in configuration file

2.5.6 Validate the solution

Perform the steps described in this section and recreate the problem. Ensure that your sessions are being maintained across requests as you expect.

If you are still experiencing problems with session affinity, go to “The next step” on page 38.

2.6 Sessions not failing over under error conditions

WebSphere Application Server allows you to configure your system so that session data is shared between the application servers in the cluster. If one of the servers fails, then a subsequent request for a session that was being handled by the failed application server can be picked up by another application server and processing continue.

There are two recommended mechanisms for handling the sharing of session data:

- ▶ The first and recommended mechanism is memory-to-memory replication. You create a session replication domain in your cluster and allow sessions to be replicated between the application servers.
- ▶ The second method is to maintain session data in database that is accessible to all servers in a cluster.

2.6.1 Collect diagnostics

To determine why sessions are not failing over, you will need to set the plug-in logging level to Trace, recreate the problem and collect the http-plugin.log and JVM logs.

Collect the following

- ▶ The plug-in log where you have set the logging level to Trace.
See 5.3, “Collecting the plug-in log” on page 91.
- ▶ JVM SystemOut and SystemErr logs
See 5.1, “Collecting JVM logs” on page 88.

2.6.2 Analyze diagnostics

You will first need to review the plug-in log. Depending on what you see, you may also need to review the JVM logs.

Plug-in trace

Refer to “Analyze the plug-in detail logging” on page 11 for details on plug-in tracing.

Look for the following:

- ▶ Review a plug-in trace to track a request that you expect to be failing over to another application server.

Example 2-13 shows the plug-in extracting the partition ID from the session cookie and finding the server that should process this request; that is, server1.

Example 2-13 Session failover, selecting the server

```
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_common:
websphereHandleSessionAffinity: Checking the partitionID in cookie JSESSIONID:
0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - DEBUG: ws_common:
websphereParseCloneID: Parsing clone ids from
'0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_common:
websphereParseCloneID: Adding clone id '34J7RVRCR5'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_common:
websphereParseCloneID: Returning list of clone ids
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupFindDwlmServer: Looking for dwlm pair
```

```
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupMatchPartitionID: Looking for partitionID
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupMatchPartitionID: Comparing curCloneID '34J7RVRCR5' to partitionID
'8I9FV6LVH'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupMatchPartitionID: Comparing curCloneID '34J7RVRCR5' to partitionID
'34J7RVRCR5'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - DEBUG: ws_server_group:
serverGroupMatchPartitionID: Match found for partitionID '34J7RVRCR5'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupGetFirstServer: getting the first server
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupGetServerByID: Comparing curCloneID '12522k3ef' to server clone id
'12522k3ef'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - DEBUG: ws_server_group:
serverGroupGetServerByID: Match for clone 'IBM99TVXRDNode1_server1'
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupFindDwlmServer: Match for clone 'IBM99TVXRDNode1_server1'
primary server
```

Example 2-14 shows the plug-in trying to connect to `server1` to maintain session affinity. The attempt fails because `server1` is down. In the message, OS error 10061 refers to the Windows TCP connection refused error.

Example 2-14 Failed connection

```
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_server_group:
lockedServerGroupUseServer: Server IBM99TVXRDNode1_server1 picked, weight 0.
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_common:
websphereFindTransport: Finding the transport
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - DETAIL: ws_common:
websphereFindTransport: Setting the transport(case 2): IBM99TVXRD.au.ibm.com on
port 9081
[Tue Apr 17 17:06:30 2007] 00001584 000014b8 - TRACE: ws_common:
websphereExecute: Executing the transaction with the app server
...
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - ERROR: ws_common:
websphereGetStream: Failed to connect to app server on host
'IBM99TVXRD.au.ibm.com', OS err=10061
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_common:
websphereGetStream: socket 644 closed - failed to connect
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - ERROR: ws_common:
websphereExecute: Failed to create the stream
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - ERROR: ws_server:
serverSetFailoverStatus: Marking IBM99TVXRDNode1_server1 down
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - STATS: ws_server:
serverSetFailoverStatus: Server IBM99TVXRDNode1_server1 : pendingRequests 0
failedRequests 1 affinityRequests 1 totalRequests 1.
```

```
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - ERROR: ws_common:
websphereHandleRequest: Failed to execute the transaction to
'IBM99TVXRNode1_server1'on host 'IBM99TVXRD.au.ibm.com'; will try another one
```

Finally, the plug-in goes back to the session cookie and checks for another available server to handle the request and resume the user session based on the partition ID as shown in Example 2-15. Note also the plug-in retrieving an updated partition table from the application server.

Example 2-15 Session failover

```
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_common:
websphereHandleSessionAffinity: Look for partitionID in affinity cookie
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_common:
websphereHandleSessionAffinity: Checking the partitionID in cookie JSESSIONID:
0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - DEBUG: ws_common:
websphereParseCloneID: Parsing clone ids from
'0001WxG_w55xP_kcCK-5-PqZt3A:34J7RVRCR5'
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_common:
websphereParseCloneID: Adding clone id '34J7RVRCR5'
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_common:
websphereParseCloneID: Returning list of clone ids
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupFindDwlmServer: Looking for dwlm pair
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - DEBUG: ws_server_group:
serverGroupNextRoundRobinServer: Round Robin load balancing
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupNextRoundRobinServer: numPrimaryServers is 2
...
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_server_group:
lockedServerGroupUseServer: Server IBM99TVXRNode2_server2 picked,
weight 0.
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupIncrementConnectionCount: Server IBM99TVXRNode2_server2 picked,
pendingConnectionCount 1 totalConnectionsCount 1.
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - DEBUG: ws_server_group:
serverGroupNextRoundRobinServer: use server IBM99TVXRNode2_server2
[Tue Apr 17 17:06:31 2007] 00001584 000014b8 - TRACE: ws_server_group:
serverGroupFindDwlmServer: Retrieve updated dwlm partition table from server
IBM99TVXRNode2_server2 (dwlmStatus = 22)
```

If you are not seeing a partition table, session replication may not be correctly configured. Go to “Session replication incorrectly configured” on page 36.

If you are not seeing the cause of the problem in the plug-in trace, then you will need to review the JVM logs.

Analyze the JVM logs

Look for the following:

- ▶ Ensure that the application servers are processing the sessions correctly. You will need to take an HTTP session trace in WebSphere Application Server.

Debugging session handling problems in WebSphere Application Server applications is outside the scope of this document.

2.6.3 Root causes

The following is a possible root cause for this issue:

- ▶ Session replication incorrectly configured

2.6.4 Session replication incorrectly configured

For the plug-in to be able to failover sessions to another application server, you need to configure session replication in the administrative console for each application server in the cluster.

Resolve the problem

Ensure there is a valid session replication domain defined and that all cluster members are configured to use this replication domain. If there is no valid session replication domain, there will be no partition table details reported in the plug-in log.

Navigate to **Servers** → **Application servers** → *server_name* → **Session management** → **Distributed environment settings**. Figure 2-8 shows that memory to memory session replication is enabled for the application server.

[Application servers](#) > [webserver1](#) > [Session management](#) > [Distributed environment](#)

Use this page to specify how session data is saved in a distributed environment. The container uses only memory-to-memory replication for distributed sessions.

Configuration

General Properties	Additional Properties
<p>Distributed sessions</p> <p> <input type="radio"/> None <input type="radio"/> Database (Supported for Web container only.) <input checked="" type="radio"/> Memory-to-memory replication </p>	<p><input type="checkbox"/> Custom tuning parameters</p>

Figure 2-8 Memory to memory session replication enabled

Figure 2-9 shows the name of the session replication domain and the replication mode. From the Distributed environment settings page in the admin console, click **Memory-to-memory replication** to review this setting.

[Application servers](#) > [webserver1](#) > [Session management](#) > [Distributed environment](#) > [Memory-to-memory replication](#)

Use this page to configure memory-to-memory replication for failure recovery.

Configuration

General Properties

* Replication domain
 ▼

* Replication mode
 ▼

Figure 2-9 Memory to memory session replication settings

All servers that are to participate in a session replication domain must be configured with the same Replication domain name.

2.6.5 Validate the solution

Ensure your system is correctly configured to enable session replication and recreate the issue. You will need to restart the application servers for any changes to take effect.

If you are still experiencing an issue, go to “The next step” on page 38.

2.7 The next step

The symptoms and problem areas included in this activity are some that you are more likely to experience. However, there are other things that can go wrong with plug-in load balancing.

2.7.1 Java heap problems

Guidance on Java heap problems can be found in the following references:

- ▶ The IBM Guided Activity Assistant tech preview for the IBM Support Assistant has a Java memory troubleshooting activity. For more information about downloading and installing this diagnostic tool, see:

<http://www-1.ibm.com/support/docview.wss?rs=180&uid=swg24011818>

- ▶ *Diagnosing out-of-memory errors and Java heap memory leaks*

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/ttrb_mdd4j.html

- ▶ *Memory leak detection and analysis in WebSphere Application Server: Part 2: Tools and features for leak detection and analysis*

http://www-128.ibm.com/developerworks/websphere/library/techarticles/0608_poddar/0608_poddar.html

2.7.2 Search online support

If you are sure the problem is with the plug-in, there are things that you can do before contacting IBM support. First, you should review the documentation that you have gathered for errors that were not addressed in this paper and search support sites for information or fixes.

Look for current information available from IBM support on known issues and resolutions using the following search argument:

<http://www-1.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCC2GP&q=MustGather>

Look also at the WebSphere Information Center documentation and other technotes for additional resources for diagnosing and fixing plug-in work load management issues:

- ▶ *Communicating with Web servers*

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/twsv_plugin.html

- ▶ *Web server plug-in configuration properties*

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/rwsv_plugin_propstable.html

- ▶ *Understanding IBM HTTP Server plug-in load balancing in a clustered environment*

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219567>

- ▶ *WebSphere plugin failover delayed when system is physically unavailable*

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21052862>

- ▶ *Understanding HTTP plug-in failover in a clustered environment*

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219808>

2.7.3 Re-evaluate the symptoms

If, after going through this process, you still have an undiagnosed problem, we recommend that you go back to *Approach to Problem Determination in WebSphere Application Server V6* at:

<http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf>

Review the problem classifications to see if there are any other components that might be causing the problem.

2.7.4 Contact IBM

There is currently no MustGather documentation for plug-in problems in WebSphere Application Server V6.1. If these steps do not resolve your problem, contact IBM technical support for assistance.

EJB workload management problem determination

The EJB Workload Management (WLM) component in WebSphere Application Server provides routing services for incoming application requests so that these can be distributed to Enterprise JavaBeans™ (EJBs). WLM also provides failover capabilities among the servers hosting the EJB. An EJB client is a process that makes a call to an EJB container to process an EJB request. The following types of EJB clients participate in EJB WLM automatically:

- ▶ Clients in the same application server (servlets, JSPs, EJBs)
- ▶ Clients in a different application server (servlets, JSPs, EJBs)
- ▶ Java applications running within a WebSphere client container
- ▶ Stand-alone Java applications using the WebSphere-supplied Java Runtime Environment (JRE™)

EJB WLM uses a weighted round robin algorithm to decide which server to send a request to. When you create a cluster, all cluster members are assigned a weight of 2 so that requests should be evenly distributed among the cluster members. You have the option of modifying the weight so that powerful servers are sent more requests for processing than less powerful servers.

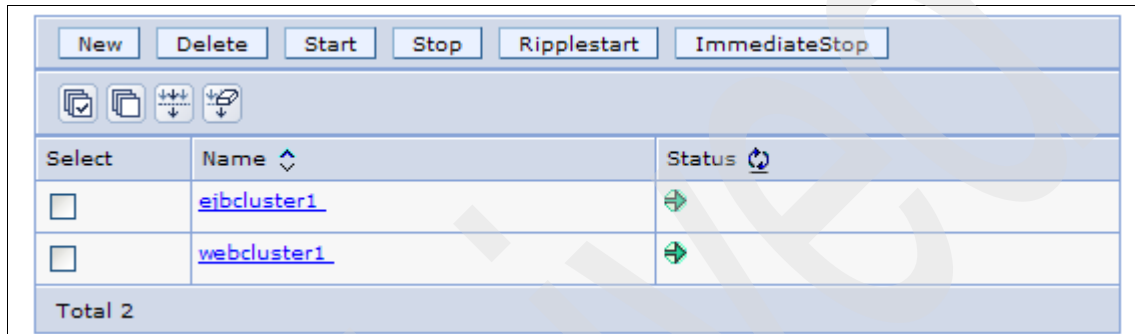
Problems with the EJB WLM can cause requests to fail or requests to be incorrectly routed. This can lead to performance degradation.

3.1 Check system integrity

If you think you might have a EJB WLM problem, the first step is to verify the system integrity:

- ▶ Verify that all cluster member servers are up and running.

In the administrative console, navigate to **Servers** → **Clusters** and ensure your clusters are started (as shown in Figure 3-1). All cluster members need to be started to ensure even load balancing.






New Delete Start Stop Ripplestart ImmediateStop		
		
Select	Name	Status
<input type="checkbox"/>	ejbcluster1	
<input type="checkbox"/>	webcluster1	
Total 2		

Figure 3-1 Started clusters

- ▶ Ensure that all the machines in your configuration have TCP/IP connectivity to each other by running the `ping` or `telnet` command:
 - From each physical server to the deployment manager
 - From the deployment manager to each physical server
 - From the client to each physical server
 - Between all physical servers
- ▶ If possible, try accessing the enterprise bean directly on the problem server to see if there is a problem with TCP/IP connectivity, application server health, or other problem not related to workload management. You will need a custom EJB client to achieve this.

3.2 Identify symptoms

The following are typical symptoms of an EJB WLM problem.

- ▶ EJB application requests do not get serviced
- ▶ EJB requests are not distributed evenly or to all servers

- ▶ Failing server still receives EJB requests (failover fails)
- ▶ Restarted servers do not share the workload

3.3 EJB application requests do not get serviced

This section discusses steps to diagnose your problem if your EJB client is making a request to an EJB that is running in a clustered environment, but that request is not getting serviced. Another symptom is that you are getting CORBA runtime exceptions returned to the client.

3.3.1 Collect diagnostics

When load balancing requests across a cluster, you do not know in advance which server will handle a given request. For this reason, you will need to review the logs from all the application servers in the cluster along with logs from the EJB client making the call.

Depending on the problem you are investigating, you should consider simplifying the problem determination process by reducing the number of servers to monitor while debugging the problem. You would do this by shutting down all but the minimum number of servers required to recreate the issue.

The following diagnostic data is useful in diagnosing the problem.

- ▶ JVM SystemOut and SystemErr logs for both the EJB client process and the EJB server processes.

See 5.1, “Collecting JVM logs” on page 88.

- ▶ First Failure Data Capture (FFDC) incident logs

The FFDC logs often provide a greater level of detail about a given exception and can be quite useful in debugging EJB failures. You should see a message written in the SystemOut.log pointing you to the specific FFDC log if one has been generated for your problem.

- ▶ Activity.log
- ▶ Network trace

You can collect it all at once, or start by collecting the JVM logs and then determining whether one or more of the other logs and traces are needed.

3.3.2 Analyze diagnostics

You first need to look at the JVM logs for both the EJB client process and the EJB server processes. These logs may not immediately show you the cause of the error but may refer you to further messages in the FFDC logs. The activity log may also give more information that will help you debug the problem.

If you cannot find the source of the problem from the logs, you may need to track the EJB request from the client to the server using a network analyzer to see if the problem is communications related.

Analyze the JVM logs

Look for the following:

- ▶ `org.omg.CORBA.NO_IMPLEMENT`: No cluster data errors on first access to EJB in a cluster.

If you see this error, go to “No cluster data” on page 47.

- ▶ `org.omg.CORBA.*` errors.

If you see this error, go to “CORBA errors” on page 48.

- ▶ You may also find that a problem accessing an EJB cluster does not get reported as such in the JVM logs. You may find messages indicating other problems that end up being an EJB WLM problem. For example: The messages shown in Example 3-1 may appear in the application server logs.

Example 3-1 JVM error messages

```
[1/05/07 15:14:22:391 EST] 0000003f ServiceLogger I
com.ibm.ws.ffdc.IncidentStreamImpl initialize FFDC0009I: FFDC opened
incident stream file
c:\IBM\WAS61\AppServer\profiles\Node1\logs\ffdc\webserver1_66126612_07.
05.01_15.14.22_0.txt
[1/05/07 15:14:23:052 EST] 0000003f ServiceLogger I
com.ibm.ws.ffdc.IncidentStreamImpl resetIncidentStream FFDC0010I: FFDC
closed incident stream file
c:\IBM\WAS61\AppServer\profiles\Node1\logs\ffdc\webserver1_66126612_07.
05.01_15.14.22_0.txt
[1/05/07 15:19:57:873 EST] 0000001f ThreadMonitor W  WSVR0605W: Thread
"WebContainer : 1" (0000003f) has been active for 695740 milliseconds
and may be hung. There is/are 1 thread(s) in total in the server that
may be hung.
```

Note that the WSVR0605W message appearing in this example is usually not associated with an EJB WLM problem but with a hung thread issue in the application server. However, the reference to the FFDC0009I and FFDC0010I

messages tell us that a FFDC incident has been logged. Refer to “Analyze the FFDC incident log” to see how reviewing this incident file identified the problem as EJB WLM related.

Analyze the FFDC incident log

The application server logs should alert you to the existence of a particular FFDC log that would contain further details of the problem. From Example 3-1 above, you can see the reference to the FFDC logs. Opening this log shows you the root cause of the problem as being no cluster members were available to service the request. This is shown in Example 3-2.

Example 3-2 Excerpt from the FFDC log

```
-----Start of DE processing----- = [1/05/07 15:14:22:371 EST] , key =  
com.ibm.ws.cluster.router.selection.NoClusterMembersAvailableException  
com.ibm.ws.cluster.router.selection.SelectionManager.getTarget 254  
Exception =  
com.ibm.ws.cluster.router.selection.NoClusterMembersAvailableException  
Source = com.ibm.ws.cluster.router.selection.SelectionManager.getTarget  
probeid = 254  
Stack Dump =  
com.ibm.ws.cluster.router.selection.NoClusterMembersAvailableException
```

To resolve this problem, go to “CORBA errors” on page 48.

There are other possible causes for this type of problem and the FFDC logs are a useful source of debugging information. If you are unable to determine the root cause of the problem from the FFDC incident log, review the activity log for errors reported around the time of the problem.

Analyze the activity log

The activity log is a binary log and should be reviewed using the Profiling and Logging perspective in either the Application Server Toolkit or Rational® Application Developer.

To view the activity log in the Application Server Toolkit:

1. Start the tool from the Windows command prompt or from the start script.
2. From the **Window** menu, choose **Open Perspective** → **Other** and select **Profiling and Logging**.
3. From the **File** menu choose **Import**. This brings up the Import dialog.
4. Select **Log file** and click **Next**.
5. On the next page choose **Add**, this brings up the Add Log File dialog.

6. In the Selected log files box, choose **IBM WebSphere Application Server activity log**.
7. Click the **Details** tab and enter the following data:
 - IBM WebSphere Application Server activity log file: path to the activity log
 - IBM WebSphere Application Server installation directory: path to the WebSphere Application Server installation
 - IBM WebSphere Application Server version: 6.x (rules)
8. Click **OK** and then click **Finish**.

The activity log can give you detailed information about EJB WLM processing. Look for errors around the time of the problem similar to those shown in Example 3-2. In this example, all cluster members are down and therefore the EJB request cannot be processed.

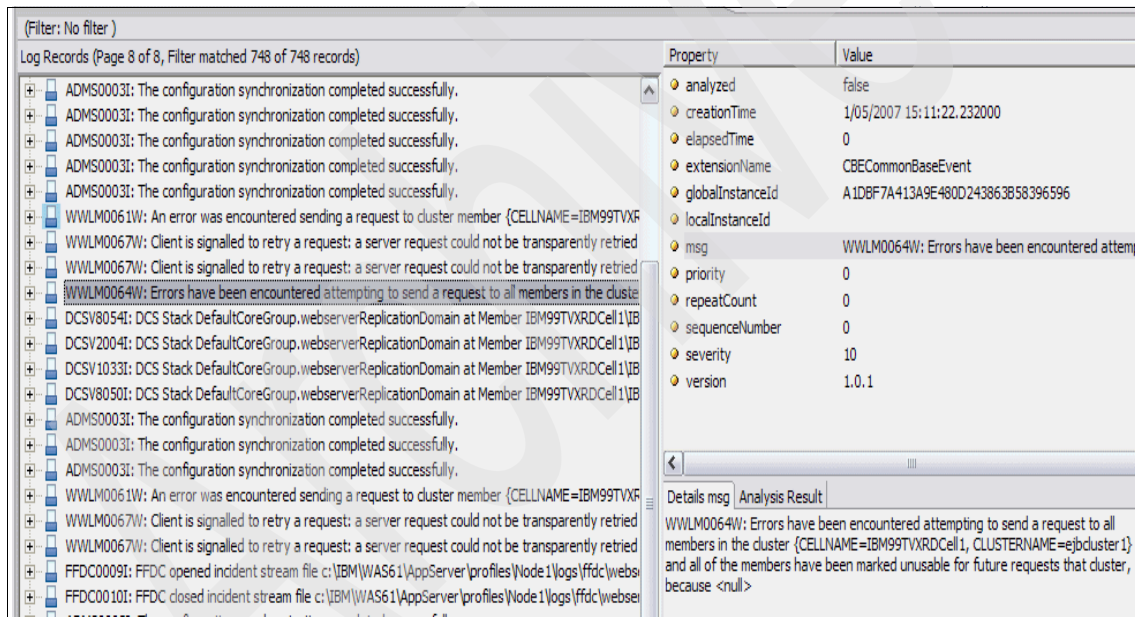


Figure 3-2 Viewing the activity log with the AST

If you can not find any indication of the problem in the activity log, take a network trace to track the EJB request and reply through the network.

Analyze the network trace

Use a tool appropriate to your operating system to collect a network trace. Some tools include snoop, tcptrace and ethereal. Refer to the following technote for further information about network tracing:

- ▶ *How to use packet trace tools iptrace, snoop, tcpdump, ethereal, and nettl*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21175744>

Recreate the problem and look for the packets that make up the EJB request. Follow the path of the request from the client to the server and follow the response back. This should show you where the request goes missing if the problem is related to network issues. Also look for the following:

- ▶ Network saturation
- ▶ Dropped or incorrectly routed packets

If you still can not resolve the problem, refer to “The next step” on page 61.

3.3.3 Root causes

The following are possible root causes for this problem:

- ▶ No cluster data
- ▶ CORBA errors

3.3.4 No cluster data

The WLM component relies on a routing table that shows which servers can handle the EJB requests. The WLM builds the routing table when the first request for an EJB is received. Depending on network topology, building this table can take some time as the Workload Manager needs to contact every server in the cluster to determine its status. Depending on the configuration, the first or first few requests can fail.

You will see the following error in the node agent logs:

org.omg.CORBA.NO_IMPLEMENT: No Cluster Data

The problem goes away when the routing table is built and normal processing continues.

Resolve the problem

Refer to the following technote for the resolution to this issue:

- *PK20304: NO_IMPLEMENT ON FIRST REQUEST TO A CLUSTER*

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg1PK20304>

3.3.5 CORBA errors

Look for the following CORBA messages:

org.omg.CORBA.TRANSIENT: SIGNAL_RETRY

This is a transient exception caused when the workload management routing service tries to route a request to a target server. This is the exception thrown to the client when there is no reply to the request.

Another EJB target may be available to service the request, but the request could not be failed over transparently by the WLM because the completion status was not determined to be "no". In this case the client application needs to determine if they want to resend the request. For example, if the EJB request updates a database table, the WLM is not able to determine if the update occurred or not as it did not get a response. Therefore this is left to the application code to determine.

Resolve the problem

You will need to determine if the application server executing the EJB is still operational. If it is not, restart the application server. If the application server is still operational then it is likely experiencing some other issue such as an application server hang or other similar error and is not a WLM issue.

Either way, the application will need to determine how to recover the transaction.

org.omg.CORBA.NO_IMPLEMENT

This exception is thrown when none of the servers participating in the workload management cluster are available. The routing service cannot locate a suitable target for the request.

For example, if the cluster is stopped or if the application does not have a path to any of the cluster members, then this exception will be thrown back to the client. There are many kinds of NO_IMPLEMENT which can be distinguished by the associated message or minor code with the NO_IMPLEMENT exception:

NO_IMPLEMENT: Retry Limit Reached or Forward Limit Reached

NO_IMPLEMENT: Retry Limit Reached and NO_IMPLEMENT: Forward Limit Reached exceptions are thrown when WLM is attempting to route a request to a server and receives an exception which is considered retryable, or the request is being forwarded to another server. In order to avoid an infinite selection loop, these exceptions will be thrown if errors are received or forwarding is done on the same server ten consecutive times.

Resolve the problem

This is likely to be an application error and should be referred to the application team for resolution. You might be able to get more detail by running a WLM trace (see “Enabling the WLM trace” on page 88).

NO_IMPLEMENT: No Available Target

This is a general exception that means the WLM may have some cluster data but perhaps not all. However with the data currently available, the WLM cannot find a valid target for the request. It is possible that cluster members have been marked unusable or just that it does not have the current data necessary to route the request to server.

Resolve the problem

Validate that all the cluster members are available by checking their status in the administrative console. Start the servers if necessary.

If possible, attempt to connect to the EJB directly on each cluster member and review the application server logs for errors that will indicate the cause of the problem.

NoAvailableTargetException

This exception is internal to IBM only, you may see it printed out in traces with the WLM trace specification enabled, but this exception is internal to the WLM code. This exception is often expected, especially in fail over and startup scenarios and if a real problem exists, it would manifest itself as one of the NO_IMPLEMENT exceptions above.

Resolve the problem

Refer this problem to IBM support as outlined below in “The next step” on page 61.

If an application server is still failing to serve requests and you are not seeing any errors, try to restart the server.

3.3.6 Validate the solution

Recreate the situation and validate that requests are being serviced. Your requests should get serviced and no CORBA related errors should appear in the logs.

If this does not resolve the issue, go to “The next step” on page 61.

3.4 EJB requests are not distributed evenly or to all servers

This section discusses steps to diagnose your problem if you are seeing EJB requests being processed, but not all cluster members appear to be processing the requests.

3.4.1 Collect diagnostics

When load balancing requests across a cluster, you do not know in advance which server will handle a given request. For this reason, you will need to review the logs from all the application servers in the cluster along with logs from the EJB client making the call.

Depending on the problem you are investigating, you should consider simplifying the problem determination process by reducing the number of servers to monitor while debugging the problem. You would do this by shutting down all but the minimum number of servers required to recreate the issue.

Collect the following diagnostic data:

- ▶ Performance Monitoring Infrastructure (PMI) data
- ▶ JVM SystemOut and SystemErr logs

See 5.1, “Collecting JVM logs” on page 88.

3.4.2 Analyze diagnostics

First determine the distribution of EJB requests among your cluster members by analyzing the EJB WLM statistics available from the PMI. If you identify a server not processing any requests or uneven load balancing, review the JVM logs to see if there are any errors or problems. You should also review server resource usage to determine if the problem is caused by server load.

Analyze the PMI data

You first need to ensure that the server is set to collect these PMI statistics.

For each application server you want to monitor:

1. Navigate to **Monitoring and tuning** → **Performance Monitoring Infrastructure (PMI)** and choose your server name.
2. Ensure that the **Enable Performance Monitoring Infrastructure (PMI)** check box is checked.
3. In the Currently Monitored Set box, click the **Custom** link.
4. In the Configuration window, expand the Workload Management tree and choose either **server**, if the application server is servicing EJB requests, or **client** if the application server is making EJB requests or both if applicable.
5. In the table on the right, select all of the available metrics and choose **Enable**.
6. Save your changes and restart the application server.

Figure 3-3 shows the workload management Client metrics enabled.

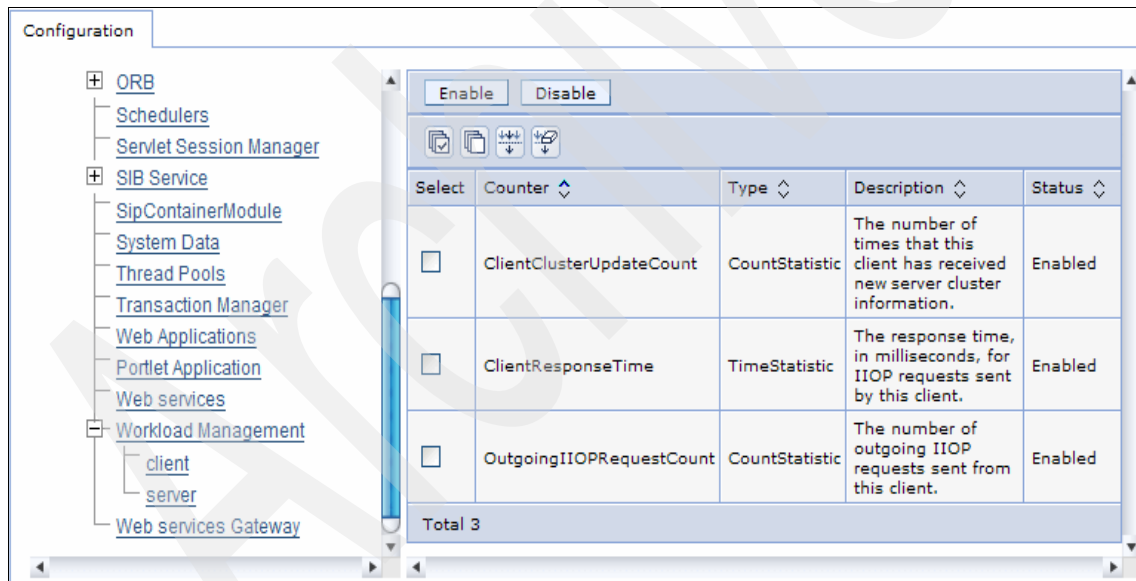


Figure 3-3 Enabling WLM PMI statistics

Recreate the issue and then review the statistics on each server that is acting as a client:

1. Navigate to **Monitoring and tuning** → **Performance viewer** → **Current activity** and click each server.

2. In the TPV window expand the Performance modules and then the Workload management trees.
3. Select the checkbox next to client and select **View modules**. You can then view the data as a graph or as a table.

Figure 3-4 shows the EJB client request counts being made from a single servlet client to a cluster of two EJB servers. You can see that 386 requests were made.

Time	client OutgoingIIOPRequestCount	client ClientResponseTime	Servlets RequestCount	Servlets ServiceTime
2:28:46 PM	386.00	33.11	20.00	851.15
2:28:13 PM	386.00	33.11	20.00	851.15
2:27:42 PM	386.00	33.11	20.00	851.15
2:27:12 PM	386.00	33.11	20.00	851.15
Total 4				

Figure 3-4 EJB WLM client requests

Next, look at the EJB server statistics for the servers running the EJBs. Figure 3-5 shows the number of requests processed by one of the two servers. You can see that it processed 152 requests, this is approximately 40% of the client requests and represents reasonably even workload balancing.

Time	server IIOPRequestCount	server WLMClientsServicedCount	server ServerResponseTime	BeenThere#BeenThere.jar CreateCount
2:25:49 PM	152.00	2.00	5.08	1.00
2:25:18 PM	152.00	2.00	5.08	1.00
2:24:47 PM	152.00	2.00	5.08	1.00
2:24:17 PM	152.00	2.00	5.08	1.00
2:23:46 PM	152.00	2.00	5.08	1.00
2:23:16 PM	152.00	2.00	5.08	1.00
2:22:45 PM	152.00	2.00	5.08	1.00
2:22:15 PM	152.00	2.00	5.08	1.00
2:21:45 PM	112.00	2.00	6.62	1.00
2:21:14 PM	100.00	1.00	6.41	1.00
2:20:44 PM	100.00	1.00	6.41	1.00
2:20:13 PM	100.00	1.00	6.41	1.00

Figure 3-5 EJB WLM server requests

Note also the WLMClientsServicedCount statistic. This shows the number of clients that are making WLM based EJB requests. In this case there are two clients.

If you are seeing uneven load distribution across the cluster members, it may be caused by:

- ▶ Unbalanced cluster member weights. For more information, see “Unbalanced cluster member weights” on page 53.
- ▶ Transaction affinity causing request distribution to be uneven. For more information, see “Transaction affinity is skewing distribution” on page 55.
- ▶ The high availability manager may be disabled for a server. For more information, see “HA Manager is disabled” on page 56.

If you are still seeing unexpected distribution, first consider that the WLM uses a weighted proportional scheme to distribute EJB requests and uses feedback mechanisms that can change the routing behavior on the fly. It reacts to various scenarios and clustered server load when making routing decisions, so it is possible that WLM can function perfectly and the requests will not be balanced exactly as you expect.

If you still believe you are experiencing an uneven load balancing problem, review the JVM logs for any errors that might be causing a server to have problems processing requests.

Analyze the JVM logs

Look for the following:

- ▶ `com.omg.CORBA` errors.
If you find any of these errors, go to “CORBA errors” on page 48.
- ▶ Look for `DCSV8050I` messages that show your application server is not in a core group view.
If you see this message, go to “EJB server not in core group view” on page 56.

3.4.3 Root causes

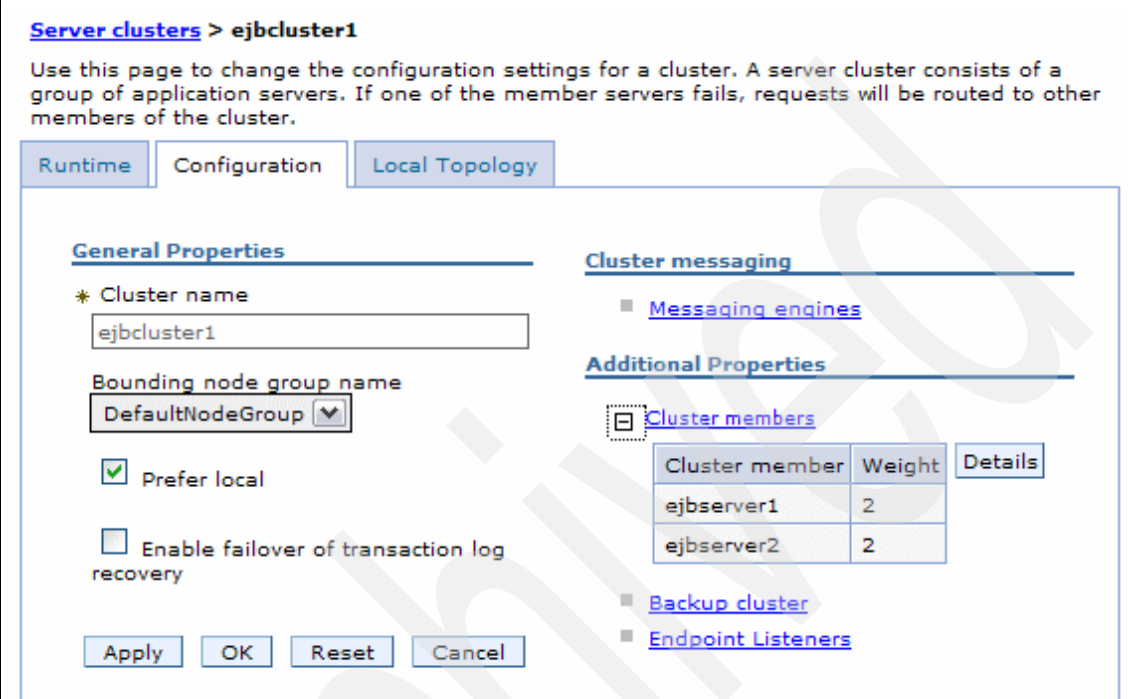
The following are possible root causes for this problem:

- ▶ Unbalanced cluster member weights
- ▶ Transaction affinity is skewing distribution
- ▶ HA Manager is disabled
- ▶ EJB server not in core group view

3.4.4 Unbalanced cluster member weights

Check the cluster member weights for the servers. These can be viewed in the administrative console as shown in Figure 3-6.

Navigate to **Servers** → **Clusters** → *clustername* and expand the Cluster members tree. The **Details** button allows you to change either the configured or runtime weights of each server in the cluster.



[Server clusters](#) > **ejbcluster1**

Use this page to change the configuration settings for a cluster. A server cluster consists of a group of application servers. If one of the member servers fails, requests will be routed to other members of the cluster.

[Runtime](#) [Configuration](#) [Local Topology](#)

General Properties

* Cluster name
ejbcluster1

Bounding node group name
DefaultNodeGroup ▼

☒ Prefer local

☐ Enable failover of transaction log recovery

[Apply](#) [OK](#) [Reset](#) [Cancel](#)

Cluster messaging

■ [Messaging engines](#)

Additional Properties

▣ [Cluster members](#)

Cluster member	Weight	Details
ejbserver1	2	
ejbserver2	2	

■ [Backup cluster](#)

■ [Endpoint Listeners](#)

Figure 3-6 EJB cluster member weights

When you modify server weights you will impact the distribution of EJB requests among the servers. For example: if you have two servers and one is more powerful than the other, you might choose to assign server weights of 7 and 2 respectively. This means that server A will be sent 7 requests to process for every 2 that are sent to server B.

EJB WLM includes “fairness” balancing so that server weights of 2 and 7 will result in the 2:7 distribution ratio with pattern like:

AAAA-B-AAA-B

Rather than a pattern like:

A-B-A-B-AAAAA

Resolve the problem

Ensure that the cluster member load balancing weights are equal or set as expected for the cluster.

3.4.5 Transaction affinity is skewing distribution

WebSphere Application Server EJB WLM supports transactional affinity. Transactional affinity means that a particular server must service all requests that comprise a particular transaction. A transaction is initiated and controlled by the client program making the EJB requests. If the application has been written so that multiple EJB requests are part of a single transaction and that transactional affinity is required, then the WLM will route requests that make up a transaction to the server that serviced the first request.

If there are a large number of requests involved in transactional affinity, this can cause uneven routing of requests. The WLM PMI data will show you how many requests were routed to a particular server due to transactional affinity. Figure 3-7 shows you the PMI data where there is no transaction affinity. Look for the StrongAffinityIIOPRequestCount field. In this example there is no transaction affinity in use.

Time	server IIOPRequestCount	server StrongAffinityIIOPRequestCount	server NoAffinityIIOPRequestCount
5:21:56 PM	711.00	0.00	711.00
5:21:30 PM	711.00	0.00	711.00
5:20:57 PM	86.00	0.00	86.00
5:20:29 PM	54.00	0.00	54.00
5:19:57 PM	8.00	0.00	8.00
5:19:26 PM	8.00	0.00	8.00
5:18:56 PM	8.00	0.00	8.00
5:18:25 PM	8.00	0.00	8.00
5:17:54 PM	8.00	0.00	8.00
Total 9			

Figure 3-7 PMI data for Transaction Affinity requests

Resolve the problem

As this is not really a problem with the workload manager but rather working as designed, there is no simple resolution. Review your application design so that you are not making large numbers of transactional affinity EJB requests.

Note: This situation is likely to only occur in a test environment and will probably resolve itself as the number of distinct users and therefore unique transactions increases.

3.4.6 HA Manager is disabled

A server can be excluded from participating in the WebSphere Application Server High Availability service. This can be done if you have decided a server or cluster does not need the services provided by the HA manager. However disabling it causes WLM to function improperly. The HA Manager is enabled in a WebSphere Application Server environment by default, however it is possible to disable it if you have determined you do not need its functionality.

Resolve the problem

Enable the HA manager service for the server and the cluster. See the following WebSphere Information Center articles for more information:

- ▶ *When to use a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_ham_required.html
- ▶ *Disabling or enabling a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_ham_enable.html

3.4.7 EJB server not in core group view

If the HA Manager is experiencing a problem where it can not determine the status of a cluster member, it will exclude that server from its core group view. If the HA Manager has excluded a server from the core group view, then the server is isolated from the rest of the cell and will not participate in EJB workload management.

Resolve the problem

Make sure the server is "in view" in reference to HA Manager and the core group the server belongs to. If the server is not in view then it may be isolated from the

rest of the cell and not seen as being available by the other servers, and subsequently the client.

There are many reasons why this might occur. For more information, see 4.2.6, “Network issues causing split views” on page 73.

3.4.8 Validate the solution

Recreate the situation and validate that requests are being distributed to the clustered servers as you expect.

If you still have problems, go to “The next step” on page 61.

3.5 Failing server still receives EJB requests (failover fails)

This section discusses steps to diagnose your problem if you are seeing that the WLM is trying to send requests to a server despite that server being down.

3.5.1 Collect diagnostics

Review the JVM logs for both the client making the request and the server that is failing or has failed.

Collect the following for both client and server:

- ▶ JVM SystemOut and SystemErr logs

See 5.1, “Collecting JVM logs” on page 88.

3.5.2 Analyze diagnostics

First review the JVM logs for the client making the EJB request. If this does not show you the root cause, then you might be able to determine what is happening from the JVM logs on the server that is failing or has failed.

JVM logs

Look for the following:

- ▶ CORBA messages indicating transaction retries or rollbacks such as:

```
org.omg.CORBA.TRANSIENT: SIGNAL_RETRY  
org.omg.CORBA::TRANSACTION_ROLLEDBACK
```

If you see either of these, go to “In-flight transaction trying to complete” on page 58.

If the server that the EJB client is attempting to execute the transaction against is down and the EJB request can not be completed, you will see CORBA errors as described in “CORBA errors” on page 48. If you have other available cluster members that should be able to service the request then you will need to take an EJB WLM trace (see “Enabling the WLM trace” on page 88).

3.5.3 Root causes

The following is a possible root cause for this issue:

- In-flight transaction trying to complete

3.5.4 In-flight transaction trying to complete

The client might have been in a running transaction with an EJB on the server that went down.

Since the transaction might have completed, failing over this request to another server could result in this request being serviced twice. Therefore the WLM puts the cluster member into Quiesce mode. While in Quiesce mode, the server will reject all incoming requests which it determines are new work, but still allow in-flight requests to complete. This is primarily designed to allow transaction work to finish as above to prevent unnecessary TRANSACTION ROLLBACK exceptions.

By default, Quiesce mode will last for a maximum of 3 minutes (this is configurable), although it is possible for a server to exit quiesce earlier if all registered components agree that it is okay to do so based on their own criteria.

Resolve the problem

If a request gets rejected by a server in Quiesce mode, WLM will be called with an `org.omg.CORBA.COMM_FAILURE` with a completion status of "no", and this request will be automatically retried by the WLM.

It might be that no servers are available to process the request. Refer to “CORBA errors” on page 48.

3.5.5 Validate the solution

Recreate the situation and validate that requests are being distributed to the clustered servers as you expect.

If you still have problems, go to “The next step” on page 61.

3.6 Restarted servers do not share the workload

This error occurs when the servers that were unavailable are not recognized by the workload manager component after they are restarted.

3.6.1 Collect diagnostics

You will need to review the JVM logs from the EJB client and the EJB server you have restarted. Collect the following for both client and server:

- ▶ JVM SystemOut and SystemErr logs
See 5.1, “Collecting JVM logs” on page 88.

3.6.2 Analyze diagnostics

Review the JVM SystemOut and SystemErr logs for the restored EJB server.

JVM logs

Look for the following:

- ▶ Ensure the server has finished starting, look for the message that indicates a server is ready to process requests:

```
[3/05/07 17:01:21:959 EST] 0000000a WsServerImpl A WSVR0001I:  
Server ejbserver1 open for e-business
```

If this message has not appeared, either wait for the server to finish startup or try restarting it again.
- ▶ Ensure the EJB has started correctly:

If the EJB fails to start, you will see exceptions that will describe what has gone wrong with the application on startup. Example 3-3 shows an EJB failing to start due to a `ClassNotFoundException`.

Example 3-3 EJB fails to start

```
[3/05/07 20:50:29:517 EST] 00000039 EJBContainerImpl E WSVR0068E: Attempt to  
start EnterpriseBean BeenThere#BeenThere.jar#BeenThereBean failed with  
exception: com.ibm.ejs.container.ContainerException: Failed to initialize  
BeanMetaData instance; nested exception is:  
java.lang.ClassNotFoundException:  
com.ibm.websphere.samples.beenthere.EJSStatelessBeenThereBeanHomeBean_271d3519
```

3.6.3 Root causes

Some possible root causes for this issue are:

- ▶ Server has not started, wait for the server to start or retry server startup
- ▶ Unusable interval has not expired

3.6.4 Unusable interval has not expired

Once the WLM has marked a server as unavailable, it waits for a period of time before trying to send a request to that server. This is called the unusable interval and it is set to 300 seconds (or 5 minutes) by default.

You can confirm that this is the problem by ensuring that the server is up and running and then wait for the unusable interval to elapse before checking to determine whether load balancing occurs as described in “Analyze the PMI data” on page 51.

Resolve the problem

You may need to either wait longer for the server to begin processing requests, or decrease the unusable interval.

You can adjust the unusable interval by setting the custom property `com.ibm.websphere.wlm.unusable.interval` to a value more suitable to your environment. The parameter is set in seconds.

This parameter is passed to the JVM as a “-D” parameter. If you are using the standalone WebSphere or Java application client as your EJB client, modify the command line to include this parameter:

```
java -Dcom.ibm.websphere.wlm.unusable.interval=150 MyApplication
```

If your client is an application server, you set this parameter on the JVM properties page. Navigate to **Servers** → **Application Servers** → *server_name* → **Java and process management** → **Process definition** → **Java Virtual Machine**. Scroll down to **Generic JVM arguments** and enter the parameter in the text box. Save your changes. You will need to restart the server for this change to take effect.

3.6.5 Validate the solution

Stop and restart the server to see if the WLM will start routing requests to the server.

If you still have problems, go to “The next step” on page 61.

3.7 The next step

The symptoms and problem areas included in this activity are some that you are more likely to experience. However, there are other things that can go wrong during with the EJB workload management.

Search online support

If you are sure the problem is with EJB WLM, there are things that you can do before contacting IBM support. First, you should review the documentation that you have gathered for errors that were not addressed in this paper and search support sites for information or fixes. Look for current information available from IBM support on known issues and resolutions on the following IBM support page using the following argument:

<http://www.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCMPEP>

Look also at the WebSphere Information Center documentation for additional resources for diagnosing and fixing EJB workload management issues:

► *Clusters and workload management*

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

► *Balancing workloads with clusters*

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

► *Tuning a workload management configuration*

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

Re-evaluate the symptoms

If, after going through this process, you still have an undiagnosed problem, we recommend that you go back to *Approach to Problem Determination in WebSphere Application Server V6* at:

<http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf>

Review the problem classifications to see if there are any other components that might be causing the problem.

Contact IBM

If these steps do not resolve your problem, then gather additional information as specified in the following MustGather document and raise a problem record with IBM. The following URL contains a list of the MustGather documentation for EJB workload management problems.

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21052165>

High availability manager problem determination

The *high availability manager* (commonly called the *HA manager*) is a component of WebSphere Application Server that enhances the availability of WebSphere Application Server singleton services such as transaction or messaging services. It provides a peer recovery mechanism for in-flight transactions or messages among clustered application servers. It also supports memory-to-memory session replication.

The HA Manager runs as a service within each WebSphere process; deployment manager, node agents, and application servers. In the event of a server failure, the HA Manager will failover any singleton service that was running on the failed server to a peer server. Examples of such a failover include the recovery of any in-flight transactions or the restarting of any messaging engines that were running on the failed server.

Problems with the HA Manager are typically identified by HA Manager related messages in the application server logs. However, they can also manifest as singleton services not being failed over when required or as failure to complete in-flight transactions from a failed server.

4.1 Determine if you need to use the HA Manager

The HA Manager consumes system resources, such as CPU cycles, heap memory, and sockets.

A *core group* is a high availability domain within a cell. It serves as a physical grouping of JVMs in a cell that are candidates to host singleton services. It can contain stand-alone servers, cluster members, node agents, or the deployment manager. Each of these run in a separate JVM. The amount of resources that the HA Manager consumes increases exponentially as the size of a core group increases. For large core groups, the amount of resources that the HA Manager consumes can become significant. If these services are not used then you can disable the HA Manager and free these resources.

You can disable the HA Manager on a given application server process if you do not use any of the following services:

- ▶ Memory-to-memory replication
- ▶ Singleton failover
- ▶ Workload management routing
- ▶ On-demand configuration routing

Do not disable the HA Manager on any administrative process (node agent or the deployment manager) unless the HA Manager is disabled on all application server processes in that core group.

If you disable the HA Manager on one member of a cluster, you must disable it on all of the other members of that cluster.

The following WebSphere Information Center article describes this in detail.

- ▶ *When to use a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_ham_required.html

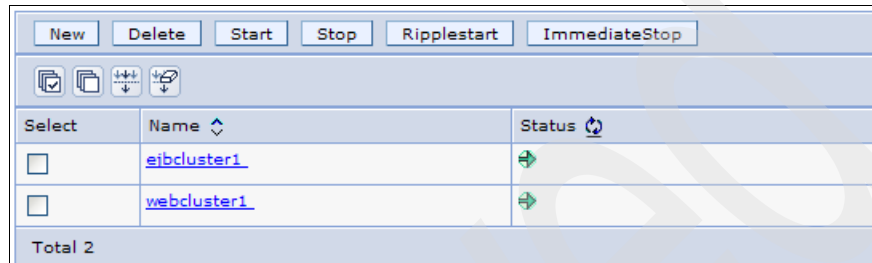
The following WebSphere Information Center article describes how to disable the HA Manager.

- ▶ *Disabling or enabling a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_ham_enable.html

4.1.1 Check system integrity

If you think you might have a problem with the HA Manager, the first step in diagnosing the problem is to verify the system integrity:

1. In the administrative console, navigate to **Servers** → **Clusters** and ensure your clusters are started (as shown in Figure 4-1). All cluster members need to be started to ensure even load balancing.



New Delete Start Stop Ripplestart ImmediateStop		
[Icons]		
Select	Name	Status
<input type="checkbox"/>	eibcluster1	➡
<input type="checkbox"/>	webcluster1	➡
Total 2		

Figure 4-1 Started clusters

2. Ensure that all the machines in your configuration have TCP/IP connectivity to each other by running the **ping** or **telnet** command:
 - From each physical server to the deployment manager
 - From the deployment manager to each physical server
 - From the client to each physical server
 - Between all physical servers
3. Ensure your core group size is not excessive.

The default core group that is created when you create a cell is made of all member servers in that cell including node agents and the deployment manager.

IBM recommends that you limit your core group size to 50 members for performance reasons. If you have large numbers of cell members, you will need to split up your core groups. For more information about this, see “Excessive core group sizes” on page 75.

4.1.2 Identify symptoms

The following are typical symptoms of an HA Manager problem.

- ▶ Singleton services not starting on failover
- ▶ Singleton services starting unexpectedly
- ▶ Server will not start

- ▶ Excessive resource usage
- ▶ HA Manager messages in the logs. These will be prefixed with HMGR, CWRLS or DCSV.

4.2 Singleton services not starting on failover

This section discusses steps to diagnose your problem if you have configured the system with a cluster of services and are using one or more singleton services, such as a messaging engine or transaction log recovery. However, you find that on server failure, these services are not being failed over.

4.2.1 Collect diagnostics

Collect the following:

- ▶ JVM SystemOut and SystemErr logs

You may need to review logs from more than just the application server that is reporting the problems. The High Availability service is cluster-aware so problems can be spread across the cluster.

Depending on the problem you are investigating, you should consider simplifying the problem determination process by reducing the number of servers to monitor while debugging the problem. You would do this by shutting down all but the minimum number of servers required to recreate the issue.

4.2.2 Analyze diagnostics

First look in the JVM logs for the messages related to servers in the cluster other than the server that has failed. You are looking for messages related to the processing of the HA manager and for a reason why it has not detected a failed server.

Look for the following:

- ▶ HMGR0140E - An event indicating that the core group membership is inconsistent was received. Recovery failed. The exception is {0}.
Go to “Inconsistent group membership” on page 67.
- ▶ HMGR1001W - An attempt to receive a message of type {0} for Agent {1} in AgentClass {2} failed. The exception is {3}

Go to “Thread pool problem” on page 73.

- ▶ DCSV8050I messages that report inconsistent view sizes across cluster members

Go to “Network issues causing split views” on page 73.

4.2.3 Root causes

The following are possible root causes for HA Manager problems:

- ▶ Inconsistent group membership
- ▶ Thread pool problem
- ▶ Network issues causing split views

4.2.4 Inconsistent group membership

Inconsistent core group membership means that the HA Manager’s view of the servers available in the core group is different from the actual servers that are available. This can occur if changes are made to the configuration that are not fully synchronized when servers are started.

Compare the core group views to the cluster topology.

Verify the size and status of the core group

Verify the size and the status of the core group by reviewing the messages in the application server logs from the member servers. One or more servers, depending on your configuration, will be elected as the *core group coordinator*. The core group coordinator is the service that monitors and maintains the core groups configured to run in the HA cluster. Example 4-1 shows the HMGR0206I message to indicate a server is the core group coordinator.

Example 4-1 Core group messages

```
[30/04/07 14:43:56:627 EST] 00000024 CoordinatorIm I   HMGR0206I: The
Coordinator is an Active Coordinator for core group DefaultCoreGroup.
...
[30/04/07 14:45:05:446 EST] 00000024 CoordinatorIm I   HMGR0207I: The
Coordinator was previously an Active Coordinator for core group
DefaultCoreGroup but has lost leadership.
[30/04/07 14:45:05:446 EST] 00000024 CoordinatorIm I   HMGR0218I: A new core
group view has been installed. The core group is DefaultCoreGroup. The view
identifier is (8:0.IBM99TVXRDCell1\IBM99TVXRDCellManager1\dmgr). The number of
members in the new view is 7.
[30/04/07 14:45:05:556 EST] 00000024 CoreGroupMemb I   DCSV8050I: DCS Stack
DefaultCoreGroup at Member IBM99TVXRDCell1\IBM99TVXRNode2\nodeagent: New view
```

```
installed, identifier (8:0.IBM99TVXRDCell1\IBM99TVXRDCellManager1\dmgr), view  
size is 7 (AV=7, CD=7, CN=7, DF=7)  
[30/04/07 14:45:05:827 EST] 00000018 ViewReceiver I DCSV1033I: DCS Stack  
DefaultCoreGroup at Member IBM99TVXRDCell1\IBM99TVXRDCellManager1\dmgr: Confirmed  
all new view members in view identifier  
(8:0.IBM99TVXRDCell1\IBM99TVXRDCellManager1\dmgr). View channel type is  
View|Ptp.
```

Example 4-1 also shows the HMGR0207I message that shows a server was a core group coordinator but has lost the leadership of the core group. Finally, it shows the messages that are produced when a new core group view is installed.

The DCSV8050I message contains the following information:

- ▶ AV = Number of Active members in the view
- ▶ CN = Members which are connected to this member. Usually, this should be the same as AV.
- ▶ CD = CN - number of suspect members
That is the number of members that the server has a connection to, but the HA Manager suspects their integrity and does not want to be in the same view with them.
- ▶ DF = Number of defined members.

If all is running as expected and all core group servers are started, then the number of active members (AV) will equal the number of defined members (DF)

A new core group view is installed every time a server that is part of the core group is stopped or started. Installing a new view might result in significant, temporary spikes in the amount of CPU consumed and the network bandwidth used.

Verify the cluster topology

Check to ensure that the cluster topologies are what you expect in the administrative console. You can review the cluster topology in the administrative console as shown in Figure 4-2. Navigate to **Servers** → **Cluster topology**. Compare this to the list of servers that the core group coordinator is aware of as shown in Figure 4-3 on page 70. Ensure that they both list the same servers.

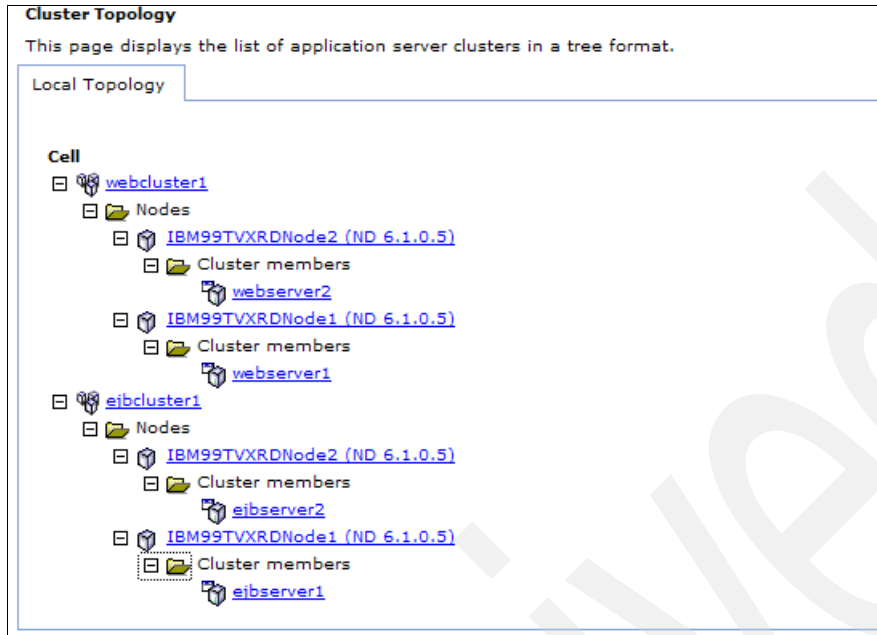


Figure 4-2 Cluster topology

Verify the HA Manager configuration

You can view the configuration of the HA Manager using the administrative console.

Navigate to **Servers** → **Core groups** → **Core group settings** → *core_group_name* → **Core group servers**.


Figure 4-3 shows the servers that the HA Manager expects to be in the core group. That is, the list of servers that the core group coordinator is aware of and connected to. Each of these servers will be a member of one or more core groups. Each core group has its own core group view, that is a view of the servers participating in the core group.



[Core groups](#) > [DefaultCoreGroup](#) > **Core group servers**

Use this page to view and manage the servers that belong to a core group. A core group server can be an application server, a deployment manager, or a node agent that is a member of a high availability core group.

⊞ Preferences

Move...

Select	Name ↕	Node ↕	Version ↕	Type ↕	Cluster Name ↕
<input type="checkbox"/>	dmgr	IBM99TVXRDCellManager1	ND 6.1.0.5	Deployment manager	
<input type="checkbox"/>	eibserver1	IBM99TVXRDCNode1	ND 6.1.0.5	Application Server	ejbcluster1
<input type="checkbox"/>	eibserver2	IBM99TVXRDCNode2	ND 6.1.0.5	Application Server	ejbcluster1
<input type="checkbox"/>	nodeagent	IBM99TVXRDCNode1	ND 6.1.0.5	Node agent	
<input type="checkbox"/>	nodeagent	IBM99TVXRDCNode2	ND 6.1.0.5	Node agent	
<input type="checkbox"/>	proxyservlet1	IBM99TVXRDCNode1	ND 6.1.0.5	Custom (PROXY_SERVER)	
<input type="checkbox"/>	webserver1	IBM99TVXRDCNode1	ND 6.1.0.5	Application Server	webcluster1
<input type="checkbox"/>	webserver2	IBM99TVXRDCNode2	ND 6.1.0.5	Application Server	webcluster1
Total 8					

Figure 4-3 Core group members

You can also review the core groups that are running at any given point in time from the administrative console.

Navigate to **Servers** → **Core groups** → **> Core group settings** → *core_group_name* and click the **Runtime** tab. Then choose **Show groups**.

Figure 4-4 shows the listing of the active core groups in the cell. These are also known as High availability groups.

<div> <div>Enable</div> <div>Disable</div> </div>				
<div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> <div> <div></div> <div></div> </div> </div>				
Select	High availability group	Quorum	Policy	Status
<input type="checkbox"/>	GN_PS=IBM99TVXRDCell1\IBM99TVXRNode1\ejbserver1,IBM_hc	Not enabled	Clustered TM Policy	
<input type="checkbox"/>	GN_PS=IBM99TVXRDCell1\IBM99TVXRNode1\webserver1,IBM_h	Not enabled	Clustered TM Policy	
<input type="checkbox"/>	GN_PS=IBM99TVXRDCell1\IBM99TVXRNode2\ejbserver2,IBM_hc	Not enabled	Clustered TM Policy	
<input type="checkbox"/>	GN_PS=IBM99TVXRDCell1\IBM99TVXRNode2\webserver2,IBM_h	Not enabled	Clustered TM Policy	
<input type="checkbox"/>	RepDomainName=dummyRepDomain,WMContextRoot=Sessionf	Not enabled	DefaultNOOPPolicy	
<input type="checkbox"/>	RepDomainName=webserverReplicationDomain,WMContextRoot:	Not enabled	DefaultNoQuorumOneOfNPoicy	
<input type="checkbox"/>	RepDomainName=webserverReplicationDomain,WMContextRoot:	Not enabled	DefaultNoQuorumOneOfNPoicy	
<input type="checkbox"/>	RepDomainName=webserverReplicationDomain,WMContextRoot:	Not enabled	DefaultNoQuorumOneOfNPoicy	
<input type="checkbox"/>	RepDomainName=webserverReplicationDomain,WMContextRoot:	Not enabled	DefaultNoQuorumOneOfNPoicy	

Figure 4-4 Active core groups

Note that core groups or servers will only appear in the Runtime tab when the associated server or servers are running.

You will typically see a lot of core groups in the Runtime tab. For example, if you have enabled HTTP session replication and have installed six Web applications, that is, with six separate context roots, then you will see six separate core groups. You will also see one core group for each server to manage the session replication cache. These will be in addition to any other core groups to manage the other singleton services.

If you click a core group name, you will see the servers that are part of that particular core group as shown in Figure 4-5. This is known as the core group view.

<input type="button" value="Enable"/> <input type="button" value="Disable"/> <input type="button" value="Activate"/> <input type="button" value="Deactivate"/>				
Select	Name	Node	Version	Status
<input type="checkbox"/>	webserver1	IBM99TVXRNode1	6.1.0.5	
<input type="checkbox"/>	webserver2	IBM99TVXRNode2	6.1.0.5	
Total 2				

Figure 4-5 Core group view

Resolve the problem

Differences between the core group views and the cluster topology can be caused by corruptions in the WebSphere Application Server configuration repository. Try the following steps to restore the configuration:

1. Perform a full resynchronization from the deployment manager to all nodes.

In the administrative console, navigate to **System Administration** → **Nodes**. Check the Select check box for all nodes except the manager node as shown in Figure 4-6. Click the **Full Resynchronize** button.

<input type="button" value="Add Node"/> <input type="button" value="Remove Node"/> <input type="button" value="Force Delete"/> <input type="button" value="Synchronize"/> <input type="button" value="Full Resynchronize"/> <input type="button" value="Stop"/>				
Select	Name	Version	Discovery Protocol	Status
<input type="checkbox"/>	IBM99TVXRDCellManager1	ND 6.1.0.5	TCP	↔
<input type="checkbox"/>	IBM99TVXRNode1	ND 6.1.0.5	TCP	↔
<input type="checkbox"/>	IBM99TVXRNode2	ND 6.1.0.5	TCP	↔
Total 3				

Figure 4-6 Full resynchronize

2. Restart all servers in the cell.
3. If the previous steps do not resolve the problem, restore the configuration from a known good backup.

WebSphere Application Server supplies the following command line tools for the backup and restore of a configuration:

- backupConfig.bat or backupConfig.sh

– restoreConfig.bat or restoreConfig.sh

4.2.5 Thread pool problem

If the HA Manager is unable to obtain a communications thread from the default thread pool, you will see the HMGR1001W. This will generally be accompanied by messages showing view sizes changing unexpectedly. Among the messages you may find the following:

```
[8/26/06 5:18:33:113 EDT] 00000095 WorkQueueMana W   TCPC0005W: TCP
Channel channel_0 could not obtain thread from thread pool <null>.
```

This problem is due to a Data Replication Service (DRS) issue, where the DRS processes are consuming all the threads from the default thread pool. As a result, the transport threads used by the HA Manager (the DCS connections) are being closed unexpectedly. This leads to instability in the core group views which can lead to unexpected failures under error conditions and excessive amounts of system resources to be consumed.

Resolve the problem

Create a separate thread pool for DCS communications so that it is isolated from the DRS communications threads.

Refer to the following technote for details:

- *Tune High Availability (HA) Manager configuration for large cell environments*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21251873>

4.2.6 Network issues causing split views

If you are seeing inconsistent view sizes across the servers in your cluster, then you are experiencing a communications issue between the cluster members.

Consider a cluster with seven defined members including the deployment manager, two node agents and four application servers. The view size in this environment would be 7 and so a normal core group view message for when all servers are running would be as shown in Example 4-2.

Example 4-2 Normal core group view

```
[2/05/07 09:27:51:576 EST] 00000021 CoreGroupMemb I   DCSV8050I: DCS Stack
DefaultCoreGroup at Member IBM99TVXRDCe111\IBM99TVXRDCe111\server1: New view
installed, identifier (22:0.IBM99TVXRDCe111\IBM99TVXRDCe111Manager1\dmgr), view
size is 7 (AV=7, CD=7, CN=7, DF=7)
```

However, if you see messages similar to those shown in Example 4-3 from server1 and Example 4-4 from server2, then you are seeing inconsistent view sizes across the cluster members.

Example 4-3 Broken core group view from server1

```
[2/05/07 09:27:51:576 EST] 00000021 CoreGroupMemb I DCSV8050I: DCS Stack
DefaultCoreGroup at Member IBM99TVXRDCe111\IBM99TVXRDNodel\server1: New view
installed, identifier (22:0.IBM99TVXRDCe111\IBM99TVXRDCe11Manager1\dmgr), view
size is 2 (AV=2, CD=2, CN=2, DF=7)
```

Example 4-4 Broken core group view from server2

```
[2/05/07 09:27:51:576 EST] 00000021 CoreGroupMemb I DCSV8050I: DCS Stack
DefaultCoreGroup at Member IBM99TVXRDCe111\IBM99TVXRDNodel\server2: New view
installed, identifier (22:0.IBM99TVXRDCe111\IBM99TVXRDCe11Manager1\dmgr), view
size is 3 (AV=3, CD=3, CN=3, DF=7)
```

Resolve the Problem

This problem is caused by a communications problem between the cluster members. Verify your communications as discussed in “Check system integrity” on page 65.

The HA Manager uses the Distributed Communications Service (DCS) for communications amongst the HA managers running in each process.

Using the loopback adapter can lead to this problem. By default, the DCS endpoint addresses use a Host field of “*” (asterisk) to indicate any host can respond on that particular port as shown in Figure 4-7.

Select	Port Name ↕	Host ↕	Port ↕	Transport Details
<input type="checkbox"/>	BOOTSTRAP ADDRESS	IBM99TVXRD.au.ibm.com	2813	No associated transports
<input type="checkbox"/>	CSIV2 SSL MUTUALAUTH LISTENER ADDRESS	IBM99TVXRD.au.ibm.com	9412	No associated transports
<input type="checkbox"/>	CSIV2 SSL SERVERAUTH LISTENER ADDRESS	IBM99TVXRD.au.ibm.com	9411	No associated transports
<input type="checkbox"/>	DCS UNICAST ADDRESS	*	9357	View associated transports

Figure 4-7 DCS address endpoint

When DCS resolves the host name of a machine that it is currently running on, in addition to the expected IP address of that host, it also gets back the loopback adapter address. This causes connectivity issues within the topology. The

loopback adapter should be disabled or removed. However, if the loopback adapter is really needed, then the solution to this problem is to change the host field for each DCS_UNICAST_ADDRESS endpoint to use a dotted IP address rather than using '*' or host names.

All WebSphere Application Server processes will need to be synchronized and restarted after making these changes.

4.2.7 Excessive core group sizes

If your core group size is very large, you could find that the core group members will use more resources maintaining the core group than processing application requests. The amount of CPU and network usage increases exponentially as more members are added to a core group and this can lead to failures in the core group configuration data. Recovery from these failures is CPU intensive which may result in paging causing further failures.

Resolve the problem

Limit core group size to 50 members.

For large topologies, create multiple smaller core groups and link these core groups with a core group bridge. Core group bridges allow servers in separate core groups to share WLM information.

Refer to the WebSphere Information Center for instructions on creating and bridging multiple core groups:

- *Creating a new core group (high availability domain)*

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

Validate the solution

After making the changes described in this section, you will need to resynchronize the changes and restart the services. Then recreate the issue and ensure the problem no longer occurs.

If you are still experiencing issues with singleton services not starting on failover, go to “The next step” on page 84.

4.3 Singleton services starting unexpectedly

This section discusses steps to diagnose your problem if you have configured the system with a cluster of services and are using one or more singleton services, such as a messaging engine or transaction log recovery. However, you find that on server failure, these services are not being failed over.

The reasons why you are seeing multiples of singleton services unexpectedly are likely to be the same as those causing singleton services not to start. Go to “Singleton services not starting on failover” on page 66 to resolve this issue.

Validate the solution

After making any changes you will need to resynchronize the changes and restart the servers. Then recreate the issue and ensure the problem no longer occurs.

If you are still experiencing issues with multiple occurrences of singleton services, go to “The next step” on page 84.

4.4 Server will not start

There are many reasons why an application server will not start. You will need to collect diagnostics to determine that the reason it is not starting is due to the HA Manager.

4.4.1 Collect diagnostics

Collect the JVM SystemOut and SystemErr logs.

4.4.2 Analyze diagnostics

If you do not see the normal “open for e-business” message associated with a started server and instead see the following message continuously logged:

```
CWRLS0030W: Waiting for HAManager to activate recovery
```

Go to “Unable to obtain lock on transaction log” on page 77.

The resolution of other startup failures are outside the scope of this document.

4.4.3 Root causes

The following is a possible root cause of the server not starting:

- Unable to obtain lock on transaction log

4.4.4 Unable to obtain lock on transaction log

This section discusses steps to diagnose your problem if the application server does not start and the following message is being logged continuously in the application server logs:

CWRLS0030W: Waiting for HAManager to activate recovery

Resolve the problem

The Transaction Manager relies on the HA Manager to assign it ownership of its transaction log file. This is true even if the “highly available transaction log” feature is not used. The Transaction Manager logs a CWRLS0030W message when it is waiting for the HA Manager to assign it ownership of its transaction log.

In order for the HA Manager to assign ownership of the transaction log to the Transaction Manager, the application server must establish itself as a member of a core group view. This requires the HA Manager to establish network connectivity between all running core group members.

Reference the following technote to resolve this issue:

- *CWRLS0030W message continuously logged and WebSphere Application Server fails to open for e-business*

<http://www.ibm.com/support/docview.wss?uid=swg21245012>

4.4.5 Validate the solution

After making the changes described in this section, you will need to synchronize the changes and restart the services. Then recreate the issue and ensure the problem no longer occurs.

If you are still experiencing issues with singleton services not starting on failover, go to “The next step” on page 84.

4.5 Excessive resource usage

The core group coordinator runs on one or more services in the core group. Under some circumstances, the core group coordinator can lead to excessive resource usage.

4.5.1 Collect diagnostics

Review the JVM logs for the cluster members to determine where the core group coordinator is running. You may also need to review PMI statistics and server metrics to measure resource usage.

- ▶ JVM SystemOut and SystemErr logs
- ▶ Verbose GC data
- ▶ Server metrics

4.5.2 Analyze diagnostics

Analyze the JVM logs

First, identify the location of the core group coordinator from the JVM logs.

Look for the following in each server's JVM log:

- ▶ HMGR0206I - The Coordinator is an Active Coordinator for core group DefaultCoreGroup.

This message indicates this server is a core group coordinator. Review the verbose GC data, PMI statistics and server metrics for this server.

- ▶ In addition, look for the following message in the logs:

HMGR0152W - CPU Starvation detected. Current thread scheduling delay is {0} seconds.

This indicates a resource problem on the server. Go to “Too much load due to HA Manager” on page 81.

If you don't see any of these symptoms, continue with the problem analysis by looking at the verbose GC data.

Verbose GC data

If your JVM heap size usage is getting close to its maximum, you are likely to see an excessive number of garbage collections being performed. Frequent garbage collections in the JVM will increase the CPU being used on the server and reduce the number of requests that can be processed.

Verbose GC can be enabled dynamically on a running server through the administrative console, see Figure 4-8.

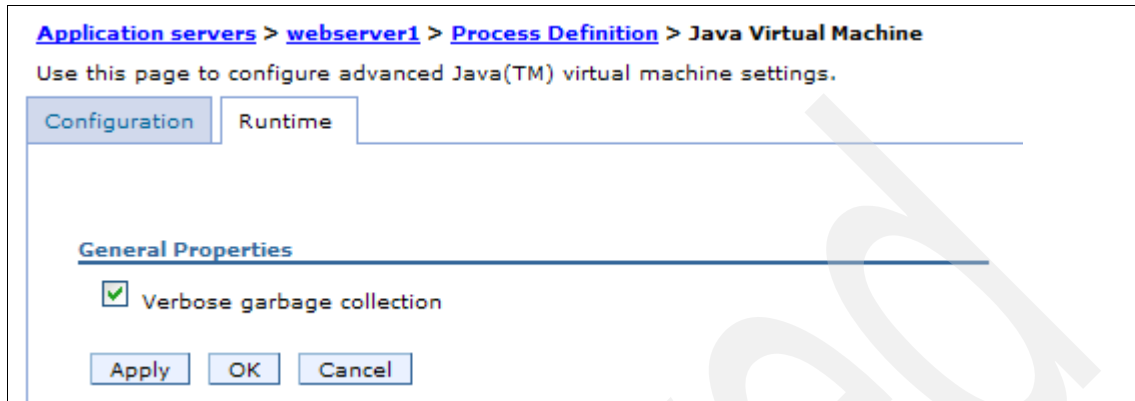


Figure 4-8 Enable verbose GC in the administrative console

Look for the following:

- Excessive garbage collection.

You can determine how frequently GC is running by looking at the `intervalms` value in the verbose GC data. This shows the number of milliseconds since the last run as shown in Example 4-5.

Example 4-5 verbose GC interval data

```
<af type="tenured" id="12" timestamp="Mon Apr 30 12:26:57 2007"
intervalms="2850.885">
```

You can also look at verbose GC data using the IBM Pattern Modeling and Analysis Tool for Java Garbage Collector (PMAT) tool that is run from the IBM Support Assistant (ISA) tool. You can use this tool to show you graphs of the GC activity.

- An OutOfMemory error

This will appear in the verbose GC output and also in the application server logs. Look for the message shown in Example 4-6.

Example 4-6 OutOfMemory in application server logs

```
[30/04/07 17:00:07:893 EST] 0000001d ServletWrapper E SRVE0068E: Uncaught
exception thrown in one of the service methods of the servlet: MyApplication. Exception
thrown : java.lang.OutOfMemoryError
```

If you are seeing this occurring only in the cluster member running the core group coordinator, go to “High JVM heap usage” on page 80.

Note that the HA Manager is only one component that uses memory in a JVM, there may be other reasons outside of the scope of this document that are causing the out of memory condition.

If you don't see any of these symptoms, continue with the problem analysis by looking at the server metrics.

Server metrics

Server metrics such as CPU, I/O and page space utilization can also show uneven or unexpected load balancing. You should use the tool most appropriate to your operating system to check these statistics. For example, in Windows you could use **Task Manager** or **Performance Monitor** while in a Unix or Linux environment, you could use **vmstat** or **top**.

Look for the following:

- ▶ Unexpectedly high CPU, I/O or page space usage on the server running the HA Manager.

If you see this condition, go to “Too much load due to HA Manager” on page 81.

4.5.3 Root causes

Some possible roots causes for this issue are:

- ▶ High JVM heap usage
- ▶ Too much load due to HA Manager

4.5.4 High JVM heap usage

The HA Manager process, and in particular, the core group coordinator uses extra resources on the application servers, node agents and deployment manager. If your JVM heap size is getting close to the limit even without considering the HA Manager components, then you could experience OutOfMemory error conditions when the HA Manager components start doing work.

Resolve the problem

If possible, increase the maximum JVM heap size on the server process to provide more heap memory for the components to use.

1. In the administrative console navigate to the process:
 - For an application server,
 Servers → **Application servers** → *server_name*
 - For a node agent,
 System administration → **Node agent** → *nodeagent*
 - For the deployment manager,
 System administration → **Deployment manager**
2. Expand the Java and process management tree and click **Process definition**.
3. Click **Java Virtual Machine**.
4. Update the Maximum Heap Size to allow for the increase in heap usage.

However, ensure you do not allocate too much memory to the JVM heap as you run the risk of either limiting native memory for the Java process or causing the system to page memory. Paging is very bad for performance.

You could also consider disabling the HA Manager processes if you do not utilize the services it provides. For more information, see “Determine if you need to use the HA Manager” on page 64.

4.5.5 Too much load due to HA Manager

If you are seeing the following message:

HMGR0152W - CPU Starvation detected. Current thread scheduling delay is {0} seconds.

This means that there is a resource constraint on the server where the message appears. Examples of this include JVM heap memory exhaustion, CPU utilization or system memory being paged.

Resolve the problem

Check the CPU usage and physical memory usage stats on the server. Physical memory paging can restrict CPU availability to the HA modules.

If the HA Manager is running on the server experiencing the problem, consider moving it to a process running on a server with more capacity. You can control the servers which can run the core group coordinator by setting the preferred coordinator servers list as shown in Figure 4-9.

1. Navigate to **Servers** → **Core groups** → **Core group settings** → *core_group_name* → **preferred coordinator servers**.

[Core groups](#) > [DefaultCoreGroup](#) > **Preferred coordinator servers**

Use this page to set up a list of core group servers on which the coordinators reside.

Configuration

General Properties

<u>Core group servers</u>		<u>Preferred coordinator servers</u>
IBM99TVXRNode2/webserver2 IBM99TVXRNode1/webserver1 IBM99TVXRNode1/ejbserver1 IBM99TVXRNode2/ejbserver2 IBM99TVXRNode1/proxyserver1	Add >> Remove <<	Move up ^ Move down v IBM99TVXRNode1/nodeagent IBM99TVXRNode2/nodeagent IBM99TVXRCellManager1/dmgr

Figure 4-9 Preferred coordinator server lists

- Specify servers that are not often stopped and restarted and that run on hosts with spare CPU and memory resources.

You could also consider disabling the HA Manager processes if you do not utilize the services it provides.

See the following article in the WebSphere Information Center:

- *When to use a high availability manager*

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

4.5.6 Validate the solution

After making the changes described in this section, you will need to synchronize the changes and restart the services. Then recreate the issue and ensure the problem no longer occurs.

If you are still experiencing issues with too much load on the server hosting the HA Manager, go to “The next step” on page 84.

4.6 HA Manager messages in the logs

In many cases, HA Manager problems are only identified by HA Manager related messages appearing in the application server logs rather than by any failure condition. A good system administrator will note these messages and take steps to resolve the underlying problems before a failure condition occurs.

4.6.1 Collect diagnostics

You will see HA Manager related messages in the application server logs.

- ▶ JVM SystemOut and SystemErr logs

4.6.2 Analyze diagnostics

Look for the following in the JVM logs:

- ▶ HMGR0140E - An event indicating that the core group membership is inconsistent was received. Recovery failed. The exception is {0}.

If you see this, go to “Inconsistent group membership” on page 67.

- ▶ HMGR1001W - An attempt to receive a message of type {0} for Agent {1} in AgentClass {2} failed. The exception is {3}

If you see this, go to “Thread pool problem” on page 73.

- ▶ DCSV8050I messages that report inconsistent view sizes across cluster members

If you see this, go to “Network issues causing split views” on page 73.

- ▶ In addition, look for the following message in the logs:

HMGR0152W - CPU Starvation detected. Current thread scheduling delay is {0} seconds.

This indicates a resource problem on the server. Go to “Too much load due to HA Manager” on page 81.

- ▶ CWRLS0030W: Waiting for HAManager to activate recovery

Go to “Unable to obtain lock on transaction log” on page 77.

4.7 The next step

The symptoms and problem areas included in this activity are some that you are more likely to experience. However, there are other things that can go wrong with the HA Manager.

4.7.1 Search online support

If you are sure the problem is with the HA Manager, there are things that you can do before contacting IBM support. First, you should review the documentation that you have gathered for errors that were not addressed in this paper and search support sites for information or fixes. Look for current information available from IBM support on known issues and resolutions on the following IBM support page using the following argument:

<http://www-1.ibm.com/support/search.wss?rs=180&tc=SSEQTP&tc1=SSCZM52>

Look also at the WebSphere Information Center documentation and technotes for additional resources for diagnosing and fixing HA Manager issues:

- ▶ *High availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_hamanager.html
- ▶ *Core group View Synchrony Protocol*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_netcomp.html
- ▶ *When to use a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_ham_required.html
- ▶ *Tune High Availability (HA) Manager configuration for large cell environments*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21251873>

4.7.2 Re-evaluate the symptoms

If, after going through this process, you still have an undiagnosed problem, we recommend that you go back to *Approach to Problem Determination in WebSphere Application Server V6* at:

<http://www.redbooks.ibm.com/redpapers/pdfs/redp4073.pdf>

Review the problem classifications to see if there are any other components that might be causing the problem.

Contact IBM

If these steps do not resolve your problem, then gather additional information as specified in the following MustGather document and raise a problem record with IBM. The following URL contains a list of the MustGather documentation for HA Manager problems.

<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21201016>

Archived



Collecting diagnostic data

This chapter provides information for collecting diagnostic data useful in diagnosing workload management problems.

5.1 Collecting JVM logs

JVM logs, often referred to as SystemOut and SystemErr logs, are created for every WebSphere Application Server process (application server, cluster member, node agent, and deployment manager). They can be found in the following locations:

- ▶ WebSphere Application Server V6.x (distributed and i5/OS)

The JVM log files are by default named SystemOut.log and SystemErr.log. The default location for the SystemOut and SystemErr logs is:

- `profile_root/logs/server_name/SystemOut.log`
- `profile_root/logs/server_name/SystemErr.log`

The location of application server logs is configurable.

- a. Select **Troubleshooting** → **Logs and Trace** in the navigation bar.
- b. Click the server name.
- c. Select **JVM logs**.

A page on the administrative console shows the location of the log file.

5.2 Enabling the WLM trace

You can enable and gather WLM trace for an EJB client or the application servers. The following sections will provide guidance on collecting the appropriate traces for WLM problems.

For more information about tracing, see the following WebSphere Information Center articles:

- ▶ *Enabling trace on client and standalone applications*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/ttrb_entrstandal.html
- ▶ *Object request broker troubleshooting tips*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/rtrb_orbcomp2.html

5.2.1 Enabling and gathering trace from a thin application client

To enable and gather trace for a thin application client, do the following:

1. Create a trace properties file by copying the `app_server_root\properties\TraceSettings.properties` file to the same directory as your client application Java archive (JAR) file.
2. Edit the properties file and change the `traceFileName` value to the location for the trace data output. For example,
`traceFileName=c:\\temp\\myAppClient.trc.`
3. Edit the properties file to remove `com.ibm.ejs.ras.*=all=enabled` and add:
`WLM*=all:ORBRas=all`
4. Add the following options to the Java command line that is used to run the client:

```
-DtraceSettingsFile=trace_properties_file  
-Djava.util.logging.manager=com.ibm.ws.bootstrap.WsLogManager  
-Djava.util.logging.configureByServer=true
```

Where *trace_properties_file* represents the name of the properties file that you created in the previous steps.

5.2.2 Enabling the trace from a J2EE application using launchClient

Use the following arguments in the `app_server_root/bin/launchclient.sh` or `.bat` file to collect a J2EE™ client trace:

- ▶ `-JVMOptions="-Dcom.ibm.CORBA.Debug=true
-Dcom.ibm.CORBA.CommTrace=true"`
- ▶ `-CCtrace=WLM*=all:ORBRas=all`
- ▶ `-CCtracefile=orbtrace.txt`
- ▶ `-CCtraceMode=basic`

For example:

```
app_server_root/bin/launchClient.sh ear_file  
-JVMOptions="-Dcom.ibm.CORBA.Debug=true -Dcom.ibm.CORBA.CommTrace=true"  
-CCtrace=ORBRas=all=enabled -CCtracefile=orbtrace.txt  
-CCtraceMode=basic
```

The ORB trace output is captured in a unique trace file named `orbtrace.txt` in the current directory.

5.2.3 Enabling the trace from the administrative console

To enable and gather the WLM trace for the application servers and WLM clients that are WebSphere application servers, do the following:

1. In the administrative console, expand **Servers** → **Application Servers** → *server_name*.
2. Select **Diagnostic Trace Service**.
3. Set **Maximum File Size** to 200 MB and select the **File radio** button in the General properties section. Set the appropriate number of historical trace files.
4. Navigate to **Change Log Detail Levels** in the Additional properties section.
5. Clear the current trace string and add the following trace string to the General properties field:
`WLM*=all:ORBRas=all`
For multiple core groups and one or more core group bridges:
`WLM*=all:ORBRas=all:Core_Group_Bridge=all`
6. Apply and save your changes.

By default, the trace is logged to a file called `trace.log` in the same location as `SystemOut.log` and `SystemErr.log`, *profile_root/logs/server_name*.

Additional information for collecting the ORB trace

In addition to the ORB tracing above, you can enable tracing of the ORB GIOP messages. This tracing is referred to by WebSphere Application Server support as a *comm trace*, and is different from the general purpose trace facility. However, you must also have enabled ORB tracing as above for comm tracing to generate any output.

1. In the administrative console, go to the ORB Service page for the application server you want to trace:
Servers → **Application servers** → *server_name* → **Container services** → **ORB service**
2. Check the box for **ORB tracing**.
3. Click **OK**, and then click **Save** to save your settings.
4. Restart the server for the new settings to take effect.

Refer to the following Information Center article for more information:

- *Enabling tracing for the Object Request Broker component*

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/rtrb_orbcomp2.html

Disabling the trace

To restore trace state back to normal, use this same process and clear the trace string. Restart the server.

5.3 Collecting the plug-in log

By default the `http-plugin.log` will be in the `logs` directory under the plug-in installation root:

`plugin_root/Plugins/logs/web_server_name/http-plugin.log`

You can determine the name and location of your plug-in log by looking in the administrative console for your Web server definition as shown in Figure 5-1. Navigate to **Web servers** → `web_server_name` → **Plug-in properties** and check the Log file name field.

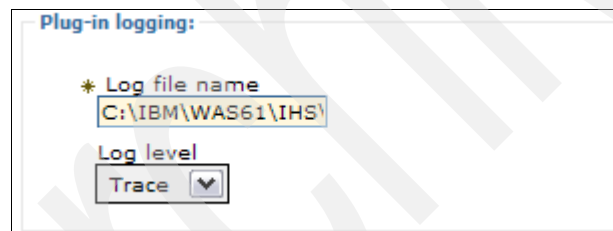


Figure 5-1 `http-plugin.log` location and log level

5.3.1 Setting the log level

The default log level for the plug-in is **Error**, this will only report error conditions such as a server being marked down.

There are several levels of logging at the plug-in that are relevant to monitoring load balancing. The logging levels are cumulative so **Stats** includes and builds on the information you get from **Warn**, and so on. The levels are:

- **Warn** - this log level will report warnings issued by the plug-in.
- **Stats** - provides basic statistics on how many requests are sent to each cluster member. It also reports on requests sent to a server to maintain

session affinity. However it does not distinguish between the requested URLs so there is no way to analyze these statistics based on URI or application.

- ▶ **Detail** - provides detailed statistics on how each request was processed and how the plug-in chose to send a given request to a particular server.
- ▶ **Debug** - similar to detail but with more information reported.
- ▶ **Trace** - provides the highest level of detail about request processing. A plug-in trace will get very large very quickly.

The logging level you should choose will depend on the complexity of your load balancing environment. If you have only one application being load balanced, then **Stats** will be sufficient to determine the load distribution. However if you have multiple applications, some with session affinity and some without, then you will need to move to a higher log level. You will probably be trying to resolve a production problem. Given this, it may be best to collect **Trace** data so that you do not have to keep running tests and traces at subsequently higher log levels to collect more information in your production environment.

Tip: If you have not enabled automatic generation and propagation of the plug-in, you will need to do this manually for the change to log level to take effect.

You do not need to restart your Web server for this change to take effect. By default, the plug-in will reload its configuration every 60 seconds. You only need wait for the reload interval to pass.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 95. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *WebSphere Application Server Network Deployment V6: High Availability Solutions*, SG24-6688
- ▶ *WebSphere Application Server V6 Scalability and Performance Handbook*, SG24-6392
- ▶ *Approach to Problem Determination in WebSphere Application Server V6*, REDP-4073

Online resources

These Web sites are also relevant as further information sources:

- ▶ WebSphere Application Server V6.1 Information Center
<http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp>
- ▶ Index of MustGather documentation
<http://www-1.ibm.com/support/docview.wss?uid=swg21145599>
- ▶ *Web server plug-in policy for WebSphere Application Server*
<http://www-1.ibm.com/support/docview.wss?uid=swg21160581>
- ▶ *Understanding IBM HTTP Server plug-in Load Balancing in a clustered environment*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219567>
- ▶ *Understanding HTTP plug-in failover in a clustered environment*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21219808>

- ▶ The IBM Guided Activity Assistant
<http://www-1.ibm.com/support/docview.wss?rs=180&uid=swg24011818>
- ▶ *Diagnosing out-of-memory errors and Java heap memory leaks*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/ttrb_mdd4j.html
- ▶ *Memory leak detection and analysis in WebSphere Application Server: Part 2: Tools and features for leak detection and analysis*
http://www-128.ibm.com/developerworks/websphere/library/techarticles/0608_poddar/0608_poddar.html
- ▶ WebSphere Application Server Product Support page
<http://www.ibm.com/software/webservers/appserv/was/support/>
- ▶ *Communicating with Web servers*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/twsv_plugin.html
- ▶ *Web server plug-in configuration properties*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/rwsv_plugin_propstable.html
- ▶ *WebSphere plugin failover delayed when system is physically unavailable*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21052862>
- ▶ *How to use packet trace tools iptrace, snoop, tcpdump, ethereal, and nettl*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21175744>
- ▶ *PK20304: NO_IMPLEMENT ON FIRST REQUEST TO A CLUSTER*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg1PK20304>
- ▶ *Clusters and workload management*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_srvgrp.html
- ▶ *Balancing workloads with clusters*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.html
- ▶ *Tuning a workload management configuration*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_tuning.html

- ▶ *When to use a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_ham_required.html
- ▶ *Tune High Availability (HA) Manager configuration for large cell environments*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21251873>
- ▶ *Creating a new core group (high availability domain)*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_newcg.html
- ▶ *CWRLS0030W message continuously logged and WebSphere Application Server fails to open for e-business*
<http://www.ibm.com/support/docview.wss?uid=swg21245012>
- ▶ *High availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_hamanager.html
- ▶ *Core group View Synchrony Protocol*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/crun_ha_netcomp.html
- ▶ *Tune High Availability (HA) Manager configuration for large cell environments*
<http://www.ibm.com/support/docview.wss?rs=180&uid=swg21251873>
- ▶ *Enabling trace on client and standalone applications*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.base.doc/info/aes/ae/ttrb_entrstanda1.html
- ▶ *Object request broker troubleshooting tips*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/rtrb_orbcomp2.html
- ▶ *Disabling or enabling a high availability manager*
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/topic/com.ibm.websphere.nd.doc/info/ae/ae/trun_ha_ham_enable.html

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

A

activity log 45–46
activity.log 43
Application Server Toolkit 45

B

backupConfig 72

C

ClassNotFoundException 59
Cluster 2
 workload management 3
cluster member weight 53
cluster members 65, 83
cluster topology 68
com.ibm.websphere.wlm.unusable.interval 60
com.omg.CORBA 53
core group 64–73, 75, 77–81, 83
core group bridge 75
core group coordinator 67, 78
core group view 53, 56, 68, 77
CPU cycles 64
CPU Starvation 83
CPU Starvation detected 81
CWRLS0030W 76–77

D

Data Replication Service (DRS) 73
DCS 73–74
DCS_UNICAST_ADDRESS 75
DCSV1033I 68
DCSV8050I 53, 67, 73–74, 83
Distributed Communications Service (DCS) 74
DRS 73

E

EJB client 3, 43
EJB client request count 52
EJB container 3
EJB server 43
EJB server statistics 52

ethereal 47

F

failover 57, 63, 66, 75, 77
fairness balancing 54
FFDC 43
FFDC log 43, 45
FFDC0009I 44
FFDC0010I 44
Forward Limit Reached 49

G

garbage collection 78–79

H

HA Manager 56, 63–69, 73–74, 76–77, 80–85
heap memory 64
heap size 80
High Availability Manager 53
highly available transaction log 77
HMGR0140E 66, 83
HMGR0152W 78, 81, 83
HMGR0206I 67, 78
HMGR0207I 67
HMGR0218I 67
HMGR1001W 66, 73, 83
HTTP session replication 71
http-plugin.log 91

I

IBM Pattern Modeling and Analysis Tool for Java
Garbage Collector (PMAT) 79
IBM Support Assistant (ISA) 79
in-flight requests 58
in-flight transactions 63
intervals 79

J

JVM heap memory exhaustion 81
JVM heap size 78, 80
JVM heap usage 80

JVM logs 21, 24, 28, 33, 43, 50, 57, 59, 66, 76, 78, 83, 88

L

launchClient 89
log level 91
loopback adapter 74

M

Maximum Heap Size 81
memory-to-memory replication 64
message prefis
 HMGR 4, 66
message prefix
 CWRLS 4, 66
 DCSV 4, 66
MustGather
 HA Manager 85
 installation 62, 85

N

network trace 43, 46
No Available Target 49
NO_IMPLEMENT 48–49
NoAvailableTargetException 49

O

online support
 EJB WLM 61
 HA Manager 84
ORB trace 90
org.omg.CORBA 57
org.omg.CORBA.* 44
org.omg.CORBA.COMM_FAILURE 58
org.omg.CORBA.NO_IMPLEMENT 44, 48
 No Cluster Data 47
org.omg.CORBA.TRANSIENT 48, 57
OutOfMemory 79–80
OutOfMemoryError 79

P

plug-in log 91
PMI data 50–51
PMI statistics 78
preferred coordinator servers 81
Profiling and Logging perspective 45

Q

Quiesce mode 58

R

Rational Application Developer 45
Redbooks Web site 95
 Contact us x
restoreConfig 73
Retry Limit Reached 49
RMI/IIOP 3
Round robin routing policy 3
routing table 47
runtime weight 54

S

server metrics 78, 80
server weight 54
server weighted routing policy 3
session replication cache 71
SIGNAL_RETRY 48, 57
singleton fail over 64
singleton service 63–64, 66, 71, 75–77
snoop 47
split views 73
SRVE0068E 79
SystemErr 21, 24, 28, 33, 43, 50, 57, 59, 66, 76, 78, 83
SystemErr log 88
SystemOut 21, 24, 28, 33, 43, 50, 57, 59, 66, 76, 78, 83
SystemOut log 88

T

TCPC0005W 73
tcptrace 47
thread pool 73
trace 88–90
traceFileName 89
TraceSettings.properties 89
transaction affinity 53, 55
transaction log 77
transaction log file 77
Transaction Manager 77
TRANSACTION_ROLLBACK 58
TRANSACTION_ROLLEDBACK 57

U

uneven load distribution 52
unusable interval 60

V

verbose GC data 78–79
view size 73

W

Web container 2–3
Web server plug-in 2–3
WLMClientsServicedCount 52
WSVR0001I 59
WSVR0605W 44



WebSphere Application Server V6.1: Workload Management Problem Determination



Redpaper™

Diagnose Web server plug-in problems

Diagnose high availability manager problems

Diagnose EJB WLM problems

This IBM Redpaper helps you to debug common problems that are related to workload management in WebSphere Application Server network deployment on distributed and on i5/OS platforms. It discusses the following areas:

- ▶ HA Manager
- ▶ EJB workload management
- ▶ Web server plug-in load balancing

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks