Monitoring GemFire Cluster

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 2: Introduction

GemFire Enterprise is a high-performance, distributed operational data management infrastructure that resides between clustered application processes and back-end data sources to provide data sharing and event distribution. GemFire cluster pools memory, CPU, network resources, and optionally local disk across multiple processes to manage application objects and behavior. It uses dynamic replication and data partitioning techniques to implement high availability, improved performance, scalability, and fault tolerance. In addition to being a distributed data container, GemFire cluster is an in-memory data management system that provides reliable asynchronous event notifications and guaranteed message delivery.

Due to the simplified architecture of the GemFire cluster and fast accessibility of the applications, GemFire cluster is gaining a rapid foothold among IT giants. As continuous access to the applications is the key in large environments, even the smallest slip in the performance of the GemFire cluster would result in huge losses. To ensure the 24x7 availability of the GemFire cluster and high performance rate, administrators need to closely monitor the performance and status of the GemFire cluster and its associated components, promptly detect abnormalities, and fix them before end-users notice. This is where eG Enterprise helps administrators.

Chapter 3: How Does eG Enterprise Monitor GemFire Cluster

eG Enterprise can monitor the GemFire cluster in an agent-based or an agentless manner. In case of the agentless approach, the remote agent used to monitor the GemFire cluster should be deployed on a remote Windows host in the environment.

3.1 Pre-Requisites for Monitoring the GemFire Cluster

- Ensure that the **gemfire.jar** file available in the **<GemFire_Install_Dir>\lib** folder is available in the **<eG_INSTALL_DIR>\agent\lib** folder.
- Regardless of the approach (agent-based or agentless), the eG agent should be configured to connect to the JRE used by the GemFire cluster so that the eG agent can pull out the required metrics, using JMX methodology.
- By default, JMX requires no authentication or security (SSL). In this case therefore, to use JMX for pulling out metrics from the GemFire cluster, the following needs to be done:
  - Login to the GemFire Cluster host.
  - The <GemFire_Install_Dir>\defaultConfigs folder used by the target application will typically contain the following files:
    - cache.xml
    - gemfire.properties
    - jmxremote.password
    - jmxremote.access
  - Edit the GemFire.properties file and set the port number at which the JMX listens to. If the JMX listens to port 1099, then set the port number as 1099.
  - Then save the file.
  - Next, during test configuration, do the following:
    - Set JMX as the mode;
    - Set the port that you defined in step 3 above (in the gemfire.properties file) as the jmx remote port;

- Set the user and password parameters to none.
- Update the test configuration.

3.2 Configuring the eG Agent to Support JMX Authentication

If the eG agent needs to use JMX for monitoring the GemFire Cluster, and this JMX requires **authentication only** (and not security), then every test to be executed by such an eG agent should be configured with the credentials of a valid user to JMX, with read-write rights. The steps for creating such a user are detailed below:

1. Login to the application host. If the application being monitored is on a Windows host, then login as a local/domain administrator to the host.
2. Go to the **<JAVA_HOME>\jre\lib\management** folder used by the target application to view the following files:
   - management.properties
   - jmxremote.access
   - jmxremote.password.template
   - snmp.acl.template
3. Copy the jmxremote.password.template file to a different location, rename it as jmxremote.password, and copy it back to the **<JAVA_HOME>\jre\lib\management** folder.
4. Open the jmxremote.password file and scroll down to the end of the file. By default, you will find the commented entries indicated by Figure 3.1 below:

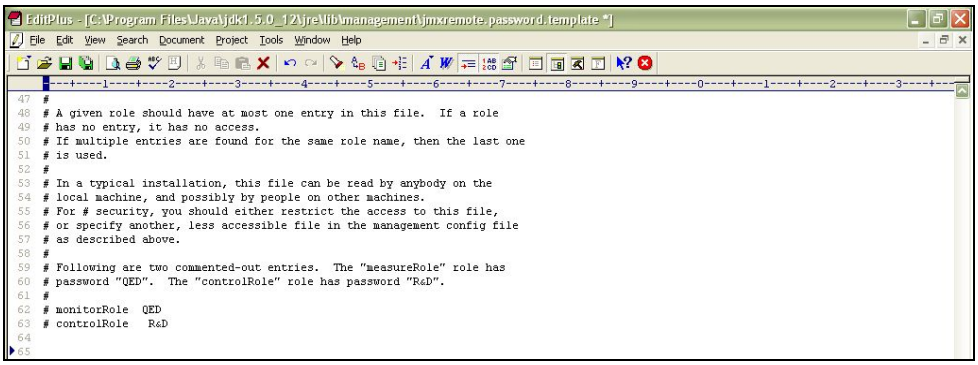

Figure 3.1: Scrolling down the jmxremote.password file to view 2 commented entries

5. The two entries indicated by Figure 3.1 are sample username password pairs with access to JMX. For instance, in the first sample entry of Figure 3.1, monitorRole is the username and QED is the password corresponding to monitorRole. Likewise, in the second line, the controlRole user takes the password R&D.

6. If you want to use one of these pre-defined username password pairs during test configuration, then simply uncomment the corresponding entry by removing the # symbol preceding that entry. However, prior to that, you need to determine what privileges have been granted to both these users. For that, open the jmxremote.access file in the editor.

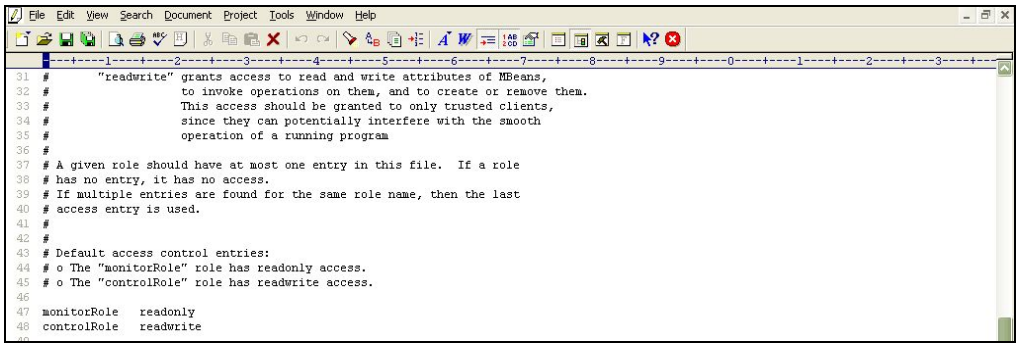

Figure 3.2: The jmxremote.access file

7. Scrolling down the file (as indicated by Figure 3.2) will reveal 2 lines, each corresponding to the sample username available in the jmxremote.password file. Each line denotes the access rights of the corresponding user. As is evident from Figure 3.2, the user monitorRole has only readonly rights, while user controlRole has readwrite rights. Since the eG agent requires readwrite rights to be able to pull out key JVM-related statistics using JMX, we will have to configure the test with the credentials of the user controlRole.

8. For that, first, edit the jmxremote.password file and uncomment the controlRole <password> line as depicted by Figure 3.3.

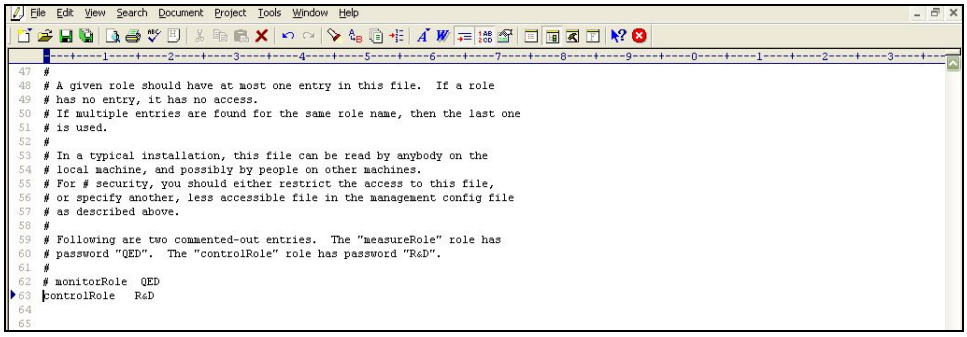

Figure 3.3: Uncommending the 'controlRole' line

9. Then, save the file. You can now proceed to configure the tests with the user name controlRoleand password R&D.
10. Alternatively, instead of going with these default credentials, you can create a new username password pair in the jmxremote.password file, assign readwrite rights to this user in the jmxremote.access file, and then configure the eG tests with the credentials of this new user. For instance, let us create a user john with password john and assign readwrite rights to john.
11. For this purpose, first, edit the *jmxremote.password* file, and append the following line (see Figure 3.4) to it:

    john john

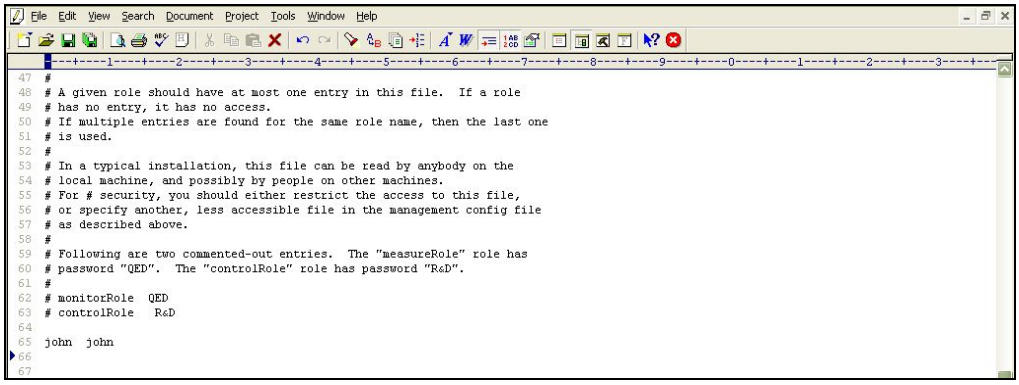

Figure 3.4: Appending a new username and password pair

12. Save the jmxremote.password file.
13. Then, edit the jmxremote.access file, and append the following line (see Figure 3.5) to it:

    john readwrite

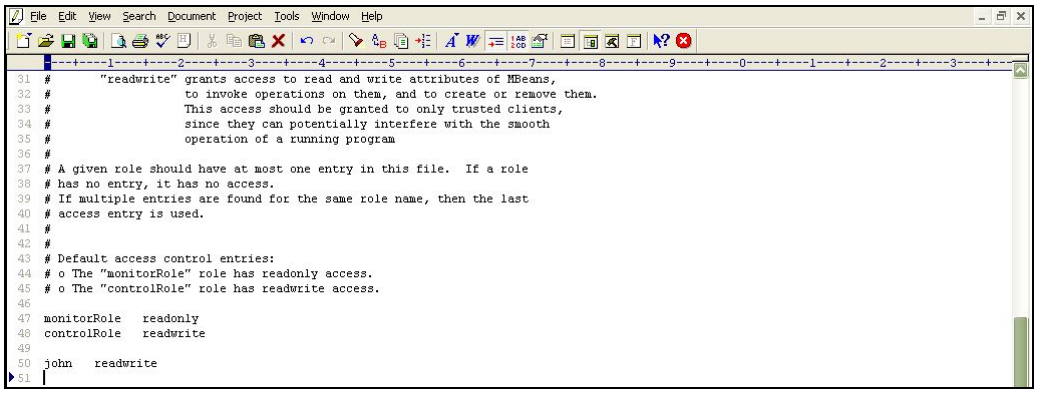

Figure 3.5: Assigning rights to the new user in the jmxremote.access file

14. Then, save the jmxremote.access file.
15. Finally, proceed to configure the tests with the user name and password, john and john, respectively.

3.2.1 Securing the 'jmxremote.password' file

To enable the eG agent to use JMX (that requires **authentication only**) for monitoring the GemFire cluster, you need to ensure that the jmxremote.password file in the **<GEMFIRE_ INSTALL_ DIR>\defaultConfigs** folder used by the target application is accessible **only by the Owner of that file**. To achieve this, do the following:

1. Login to the Windows host as a local/domain administrator.
2. Browse to the location of the jmxremote.password file using Windows Explorer.
3. Next, right-click on the jmxremote.password file and select the **Properties** option (see Figure 3.6).

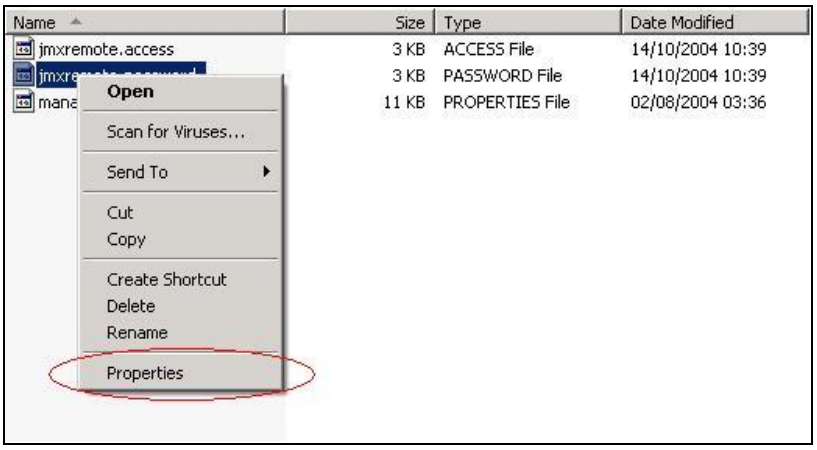

Figure 3.6: Selecting the Properties option

4. From Figure 3.7 that appears next, select the **Security** tab.

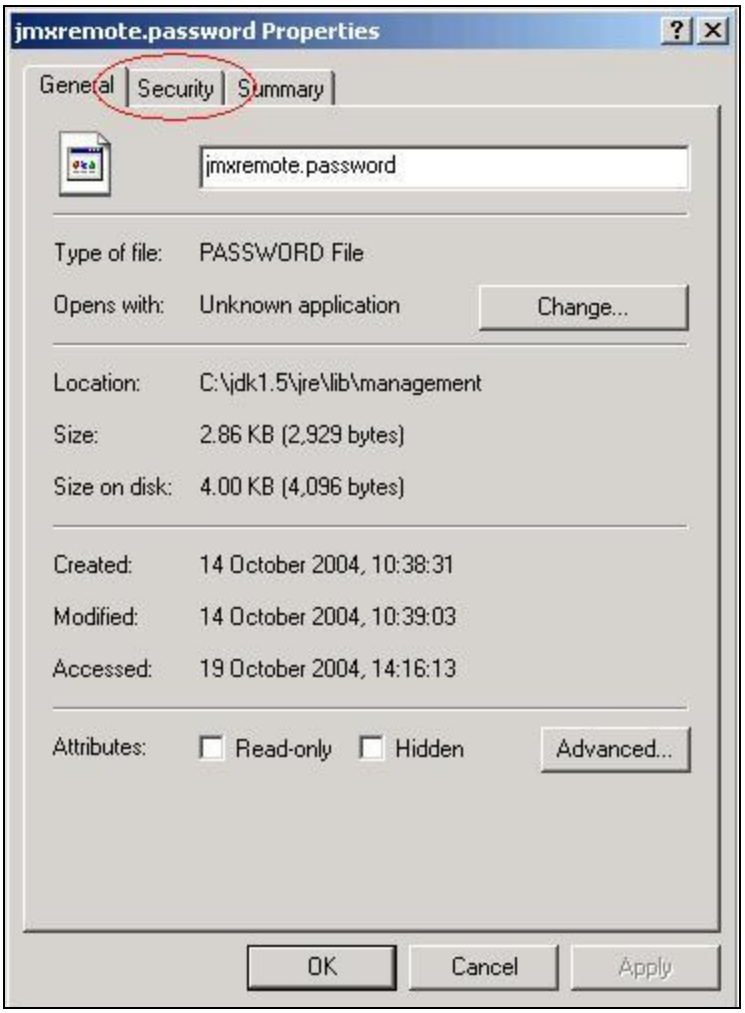

Figure 3.7: The Properties dialog box

However, if you are on a Windows computer that is not part of a domain, then the **Security** tab may be missing. To reveal the **Security** tab, do the following:

- Open Windows Explorer, and choose **Folder Options** from the **Tools** menu.
- Select the **View** tab, scroll to the bottom of the **Advanced Settings**section, and clear the check box

next to **Use Simple File Sharing**.

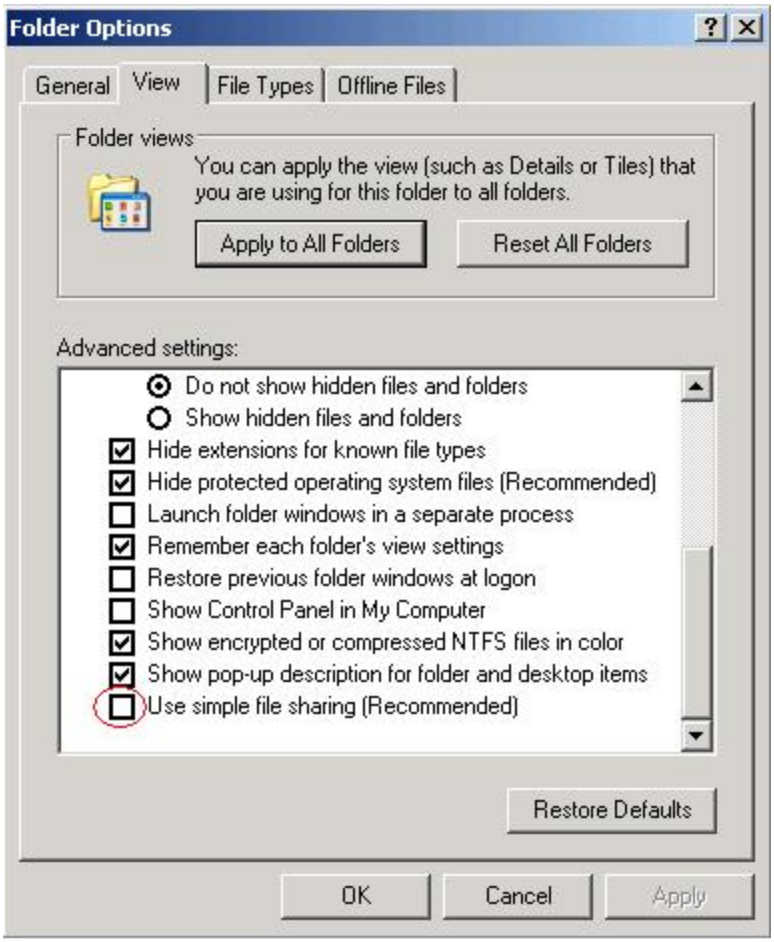

Figure 3.8: Deselecting the 'Use simple file sharing' option

- Click **OK** to apply the change
- When you restart Windows Explorer, the **Security** tab would be visible.

5. Next, select the **Advanced** button in the **Security** tab of Figure 3.9.

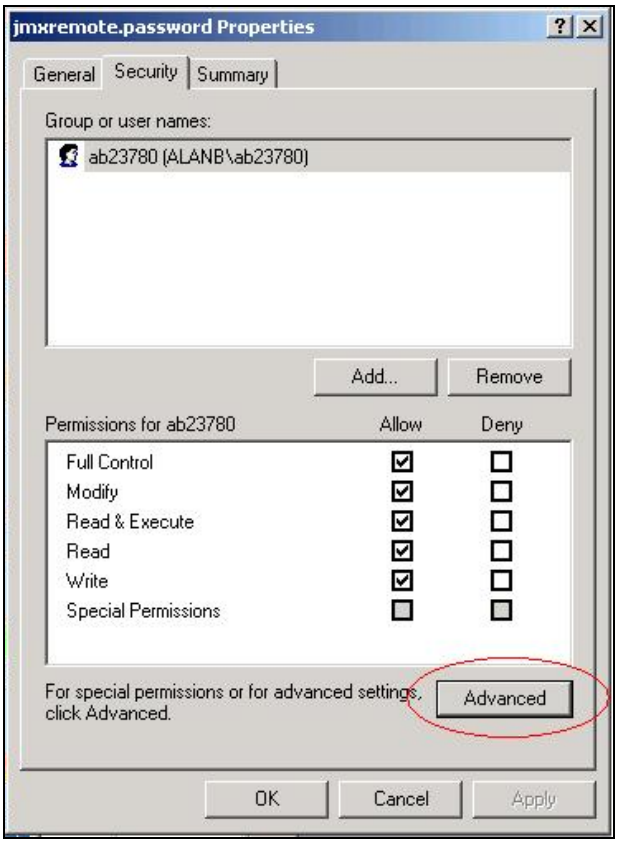

Figure 3.9: Clicking the Advanced button

6. Select the **Owner** tab to see who the owner of the file is.

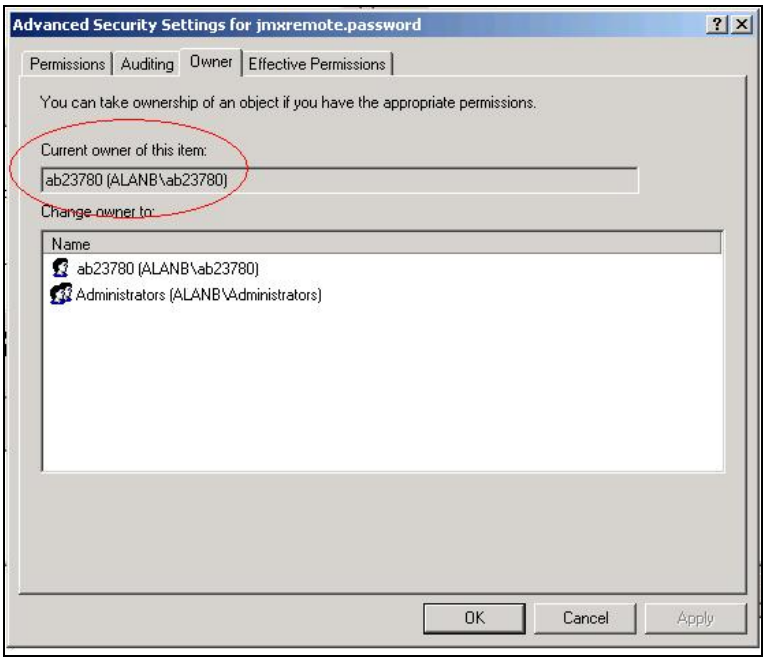

Figure 3.10: Verifying whether the Owner of the file is the same as the application Owner

7. Then, proceed to select the **Permissions** tab in Figure 3.10 to set the permissions. If the jmxremote.password file has inherited its permissions from a parent directory that allows users or groups other than the **Owner** to access the file, then clear the **Inherit from parent the permission entries that apply to child objects** check box in Figure 3.11.

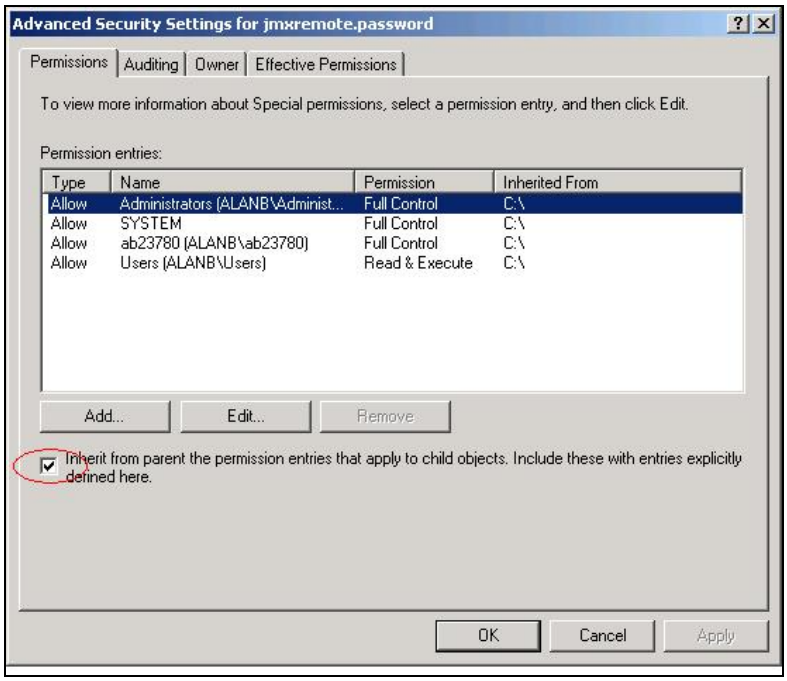

Figure 3.11: Disinheriting permissions borrowed from a parent directory

8. At this point, you will be prompted to confirm whether the inherited permissions should be copied from the parent or removed. Press the **Copy** button in Figure 3.12.

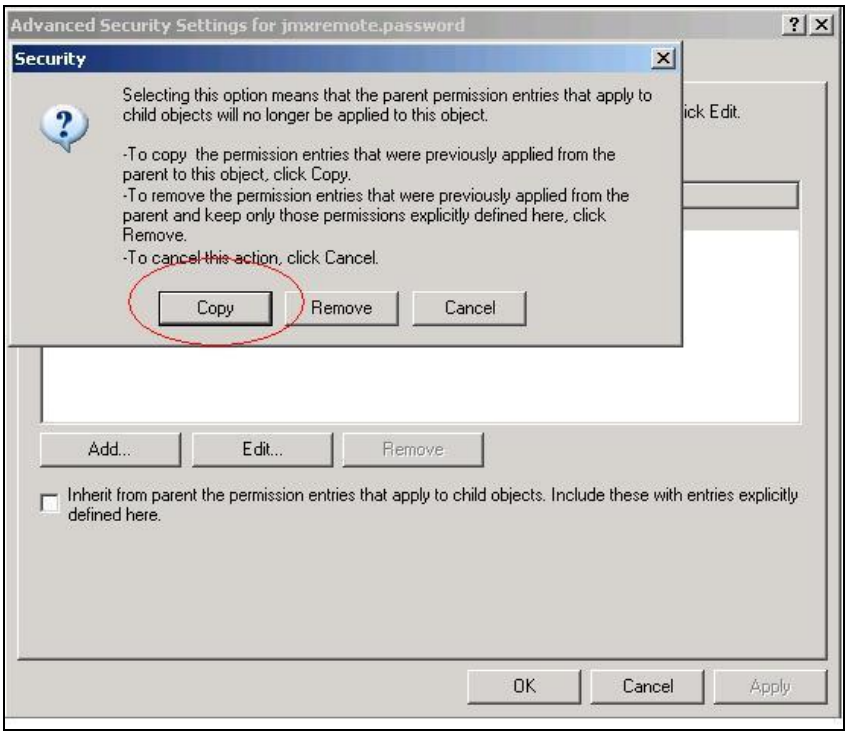

Figure 3.12: Copying the inherited permissions

9. Next, remove all permission entries that allow the jmxremote.password file to be accessed by users or groups other than the file **Owner**. For this, click the user or group and press the **Remove** button in Figure 3.13. At the end of this exercise, only a single permission entry granting **Full Control** to the owner should remain in Figure 3.13.

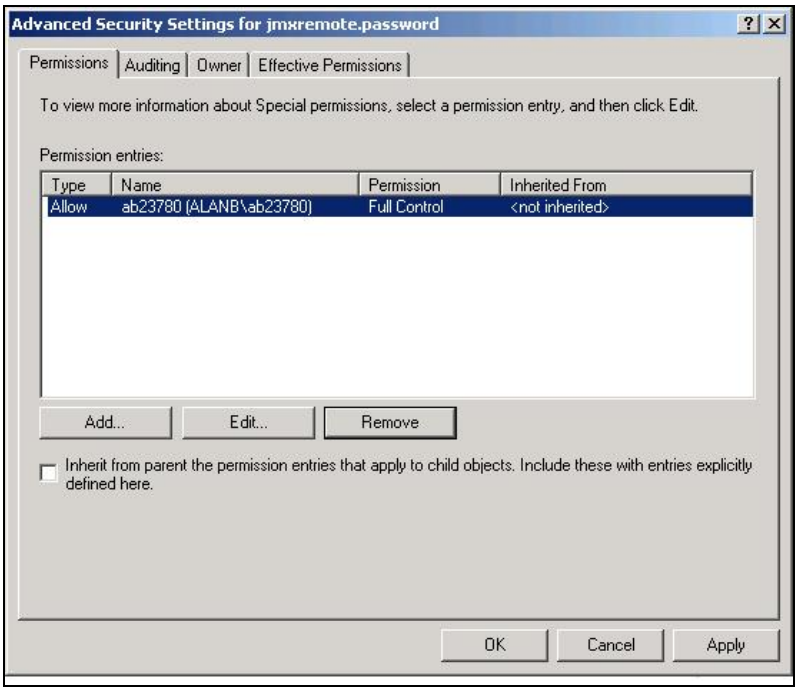

Figure 3.13: Granting full control to the file owner

10. Finally, click the **Apply** and **OK** buttons to register the changes. The password file is now secure, and can only be accessed by the file owner.

Note:

If you are trying to enable JMX on a Linux host, you might encounter issues with the way hostnames are resolved.

To solve it you might have to set the **-Djava.rmi.server.hostname=<hostname or localhost or ip>** property in the startup script of the target server.

If you are in local, simply try with **- Djava.rmi.server.hostname=localhost** or **-Djava.rmi.server.hostname=127.0.0.1**.

3.3 Managing the GemFire Cluster

The eG Enterprise cannot automatically discover the GemFire Cluster. Therefore, you need to manually add the component for monitoring. Remember that the eG Enterprise automatically manages the components that are added manually. To manage a GemFire Cluster component, do the following:

1. Log into the eG administrative interface.
2. Follow the Components -> Add/Modify menu sequence in the **Infrastructure** tile of the **Admin** menu.
3. In the **COMPONENT** page that appears next, select *GemFire Cluster* as the **Component type**. Then, click the **Add New Component** button. This will invoke Figure 3.14.

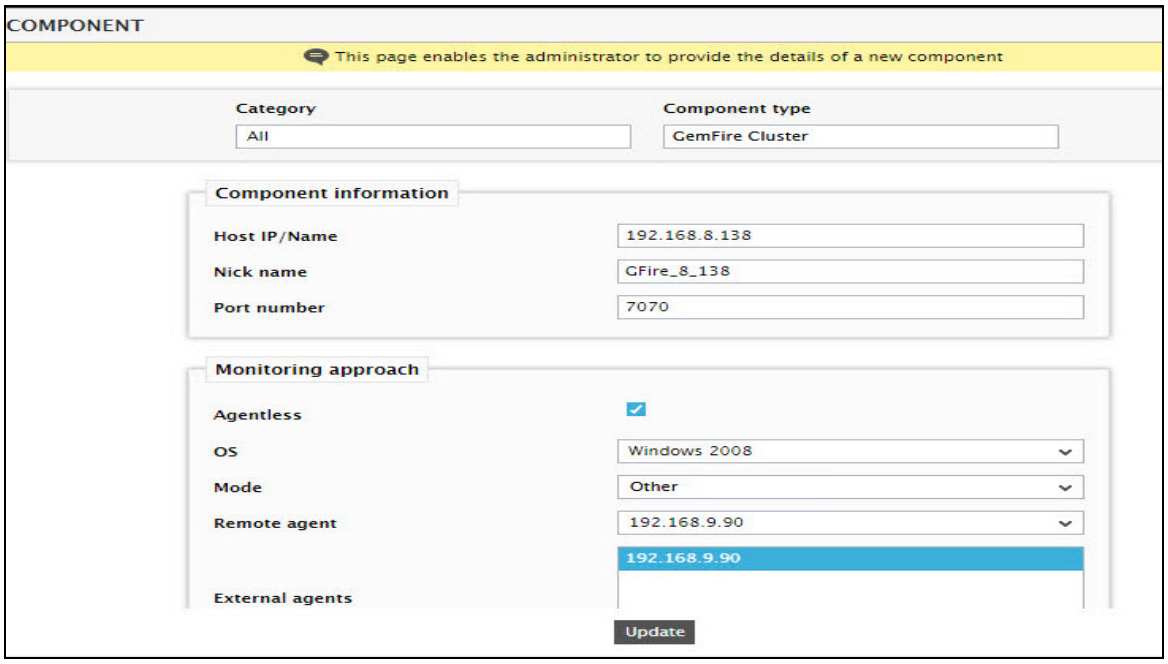

Figure 3.14: Adding a new GemFire Cluster

4. Specify the **Host IP/Name** and **Nick name** for the GemFire Cluster. Then, click on the **Add** button to register the changes.
5. Now, when you attempt to sign out of the eG administrative interface, Figure 3.15 appears, listing the tests that need to be configured for this component.

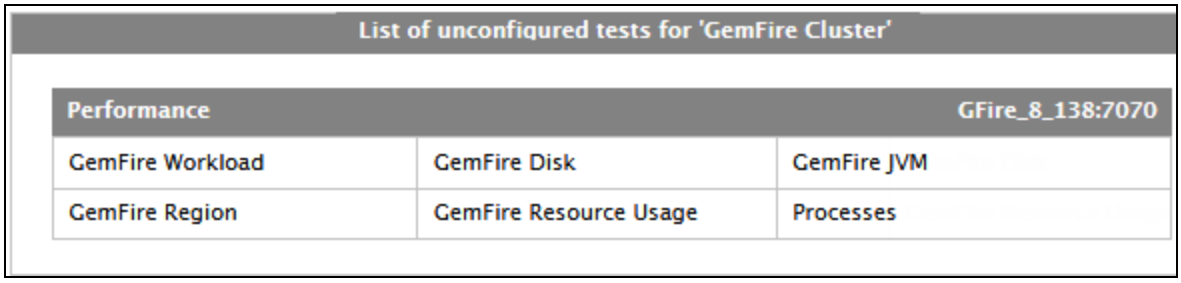

Figure 3.15: The list of unconfigured tests

6. Click on any test in the list of unconfigured tests (see Figure 3.15) to configure it. For instance, click on the **GemFire Disk** test to configure it. Figure 3.16 will then appear.

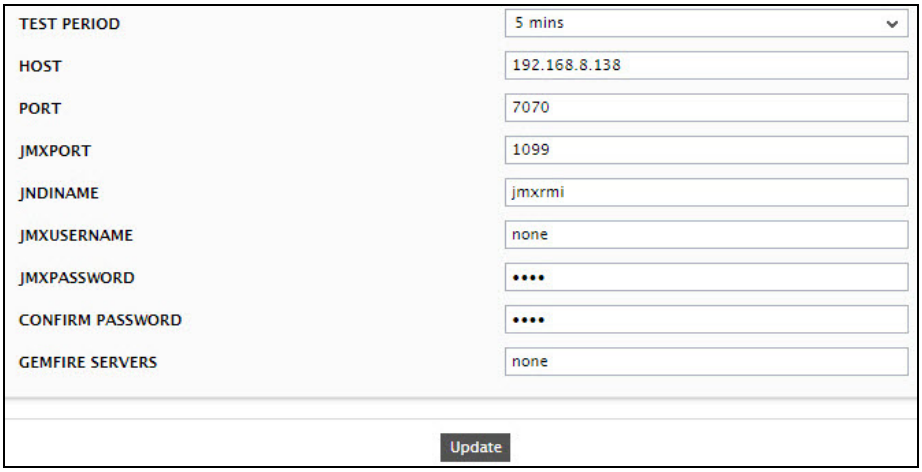

Figure 3.16: Configuring the GemFire Disk test

7. To know how to configure these tests, refer to **Monitoring GemFire Cluster using eG Enterprise**.

8. Finally, signout of the eG administrative interface.

Chapter 4: Monitoring GemFire Cluster using eG Enterprise

eG Enterprise offers an exclusive GemFire Cluster monitoring model (see Figure 4.1), which keeps tabs on the availability, I/O processing capability, the read/write latency of the disks, the JVM related statistics of each disk of the server, detects abnormalities, and helps administrators take remedial measures in due course.

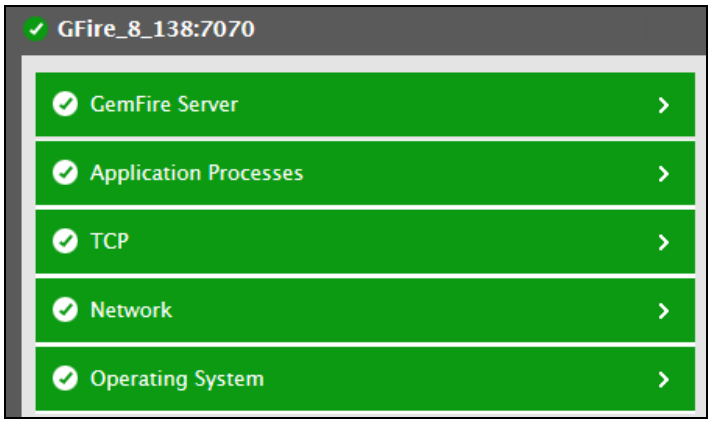

Figure 4.1: The layer model of the GemFire Cluster

Each layer of Figure 4.1 is mapped to a wide variety of tests that report a wealth of performance tests using which administrators can find quick and accurate answers to the following performance queries:

- How many cache members are associated with each GemFire region?
- How well data is read from and written to each region?
- How well data is read from and written to the disks associated with each region?
- How well data is read from and written to each disk of the target GemFire cluster?
- What is the average time taken to flush data from each disk?
- What is the total amount of data utilized by each disk?
- How many times the JVM of each disk was suspended?
- How many threads are currently in use in the JVM of each disk?
- What is the percentage of heap memory utilized on the JVM of each disk?
- How many garbage collection operation were performed on the JVM of each disk?
- What is the time taken to perform the garbage collection operation on the JVM of each disk?

- What is the amount of heap memory allocated to the JVM of each disk?
- Is each member of each resource running?
- Was the member of the resource rebooted?
- How well the CPU of each member belonging to a resource was utilized?
- How many regions were present in the cache allocated to each member?
- How well data was read from and written to the cache allocated to each member?
- What is the average rate at which data was read from and written to the cache allocated to each member?
- How many clients were connected to each member?
- How many backend servers were communicating with the GemFire server?
- What is the maximum number of connections allowed on the server?
- What is the maximum number of threads allowed to service user requests?
- What is the maximum time taken by the server to cater to user requests?
- How many virtual machines are connected to the GemFire cluster?
- How many connections are waiting for a thread?
- How many connections failed to connect to the GemFire cluster?

Since the Network layer has been discussed extensively in *Monitoring Unix and Windows Servers* document, let us now discuss the tests pertaining to the remaining layers mentioned in Figure 4.1 in detail in the forthcoming sections.

4.1 The GemFire Server Layer

Using this layer, administrators can figure out the availability, I/O processing capability, the read/write latency of the disks, the JVM related statistics of each disk of the server etc and figure out the discrepancies, if any.

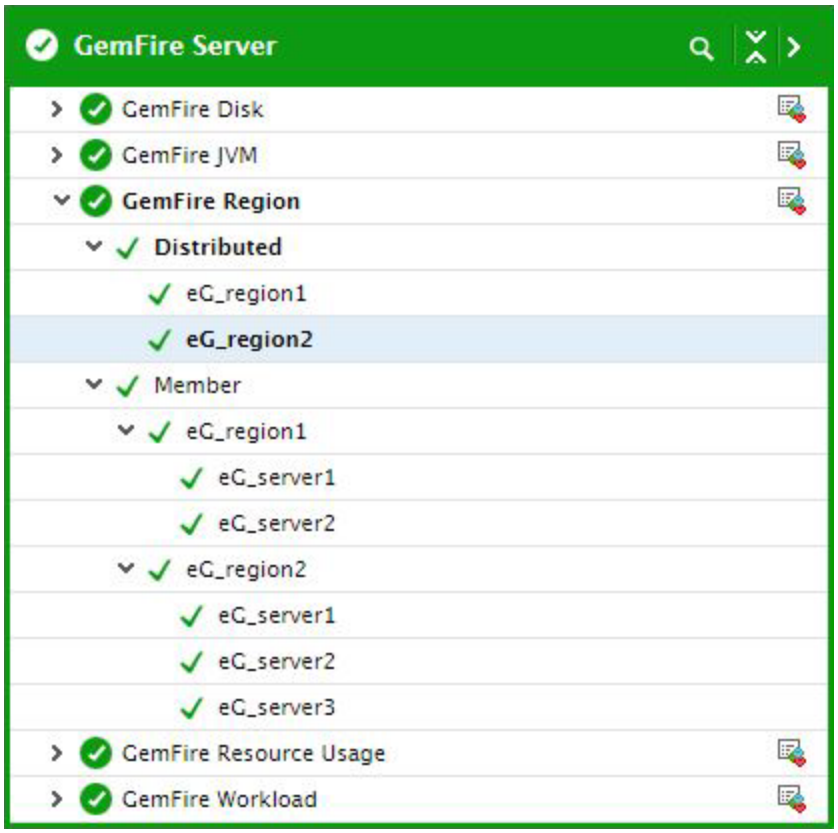

Figure 4.2: The tests pertaining to the GemFire Server layer

4.1.1 GemFire Disk Test

This test auto-discovers the disks on the target GemFire Cluster and for each disk, this test reports the rate at which data is read from and written to the disk and the time taken to flush the data into each disk. Using this test, administrators can identify the error-prone disks that may fail any time and be alerted to potential disk failures. In addition, this test points administrators to the disk from which the maximum amount of data is utilized.

Target of the test : A GemFire Cluster

Agent deploying the test : An internal/remote agent.

Outputs of the test : One set of results for each disk of the target GemFire Cluster being monitored.

Configurable parameters for the test

Parameters	Description
Test Period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port on which the specified host listens. By default, this is 7070.
JMXPort	Here, specify the port at which the JMX listens for requests from remote hosts. Ensure that you specify the same port that you configured in the management.properties file in the <JAVA_HOME>\jre\lib\management folder used by the target application.
JNDIName	The JNDIname is a lookup name for connecting to the JMX connector. By default, this is jmxrmi. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
JMX User Name and JMX Password	If JMX requires **authentication only (but no security)**, then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication.
Confirm Password	Confirm the Password by retyping it in this text box.
GemFire Servers	Specify the exact name of the server if the monitored GemFire Cluster belongs to a cluster. By default, this is *none*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Disk flush average latency	Indicates the average time taken to flush the data from this disk.	Milliseconds	A high value for this measure is a cause of concern as this may indicate slowness of the disk.
Read rate	Indicates the rate at which data was read from this disk.	Reads/sec	Comparing the value of these measures across the disks helps you in identifying the disk that is busy in terms of disk reads and disk writes.
Write rate	Indicates the rate at which data was written to this disk.	Writes/sec	
Total disk usage	Indicates the total amount of data utilized from this disk.	MB	

4.1.2 GemFire JVM Test

This test continuously monitors the JVM heap usage and promptly alerts administrators when memory usage is at the maximum. In addition, this test helps administrators figure out the statistics related to the garbage collection of the JVM.

Target of the test : A GemFire server

Agent deploying the test : An internal/remote agent.

Outputs of the test : One set of results for each disk of the target GemFire server being monitored.

Configurable parameters for the test

Parameters	Description
Test Period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port on which the specified host listens. By default, this is 7070.
JMXPort	Here, specify the port at which the jmx listens for requests from remote hosts. Ensure that you specify the same port that you configured in the gemfire.properties file in the <GEMFIRE_INSTALL_DIR>\defaultConfigs folder used by the target application.
JNDIName	The JNDIname is a lookup name for connecting to the JMX connector. By default, this is jmxrmi. If you have registered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
JMX UserName and JMX Password	If JMX requires authentication only (but no security), then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication.
Confirm Password	Confirm the Password by retyping it in this text box.
GemFire Servers	Specify the name of the server if the monitored GemFire server belongs to a cluster. By default, this is *none*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
JVM pauses	Indicates the number of	Number	

Measurement	Description	Measurement Unit	Interpretation
	times the JVM of this disk suspended i.e., paused the application which may or may not include full garbage collection.		
Number of threads in use	Indicates the total number of threads that are currently in use in the JVM of this disk.	Number	
Used heap size	Indicates the amount of heap memory that is currently utilized by the JVM of this disk.	MB	A low value is desired for this measure. Comparing the value of this measure across the JVM of the disks reveals the JVM of the disk that is more frequently utilized and is in need of additional resources.
Free heap size	Indicates the amount of heap memory that is currently available for use on the JVM of this disk.	MB	A high value is desired for this measure.
Total heap size	Indicates the total amount of heap memory on the JVM of this disk.	MB	
Heap usage	Indicates the percentage of heap memory utilized on the JVM of this disk.	Percent	A low value is desired for this measure. Comparing the value of this measure across the JVM of the disks reveals the JVM of the disk that is more frequently utilized and is in need of additional resources.
Heap free	Indicates the percentage of heap memory that is already utilized on the JVM of this disk.	Percent	A high value is desired for this measure.
Garbage collections	Indicates the number of garbage collection operation was performed on the JVM of this disk.	Number	

Measurement	Description	Measurement Unit	Interpretation
Garbage collection time	Indicates the time taken to perform the garbage collection operation on the JVM of this disk.	MilliSeconds	Ideally, the value of this measure should be low. This is because, the garbage collection (GC) activity tends to suspend the operations of the application until such time that GC ends. Longer the GC time, longer it would take for the application to resume its functions. To minimize the impact of GC on application performance, it is best to ensure that GC activity does not take too long to complete.
Allocated heap size	Indicates the amount of heap memory that was allocated to the JVM of this disk.	MB	

4.1.3 GemFire Region Test

Caches are an abstraction that describe a node in a GemFire distributed system. Application architects can arrange these nodes in peer-to-peer or client/server topologies.

Within each cache, you define data regions. Data regions are analogous to tables in a relational database and manage data in a distributed fashion as name/value pairs. A replicated region stores identical copies of the data on each cache member of a distributed system. A partitioned region spreads the data among cache members. After the system is configured, client applications can access the distributed data in regions without knowledge of the underlying system architecture. You can define listeners to create notifications about when data has changed, and you can define expiration criteria to delete obsolete data in a region.

Within a GemFire distributed system, you can define distributed regions and non-distributed regions, and you can define regions whose data is spread across the distributed system, and regions whose data is entirely contained in a single member.

Following are the different types of regions that are available for a GemFire Distributed System:

- **Partitioned:** Data is divided into buckets across the members that define the region. For high availability, configure redundant copies so each bucket is stored in multiple members with one

member holding the primary.

- **Replicated (distributed):** Holds all data from the distributed region. The data from the distributed region is copied into the member replica region. Can be mixed with non-replication, with some members holding replicas and some holding non-replicas.
- **Distributed non-replicated:** Data is spread across the members that define the region. Each member holds only the data it has expressed interest in. Can be mixed with replication, with some members holding replicas and some holding non-replicas.
- **Non-distributed (local):** The region is visible only to the defining member.

If the regions are not updated frequently or if the regions store outdated data, users accessing the data through those regions may be exposed to obsolete data. Therefore, it is important to monitor the regions that store the data in the GemFire Distributed Cache round the clock! The **GemFire Region** test helps administrators in this regard!

This test auto-discovers the regions of the GemFire Distributed system and reports whether the gateway through which data is sent from each region is enabled, the number of cache members associated with each region, captures the bucket count and rate at which data is added to each region, and measures the level of disk I/O activity on each region, so that administrators are notified of region related problems well before they impact the entire distributed system.

Target of the test : A GemFire Cluster

Agent deploying the test : An internal/remote agent.

Outputs of the test : One set of results for each Cache Member:Region of the target GemFire server being monitored

Configurable parameters for the test

Parameters	Description
Test Period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port on which the specified host listens. By default, this is 7070.
JMXPort	Here, specify the port at which the jmx listens for requests from remote hosts. Ensure that you specify the same port that you configured in the management.properties file in the <JAVA_HOME>\jre\lib\management folder used by the target application
JNDIName	The JNDIname is a lookup name for connecting to the JMX connector. By default, this is jmxrmi. If you have resgistered the JMX connector in the RMI registery using a

Parameters	Description
	different lookup name, then you can change this default value to reflect the same.
JMX UserName and JMX Password	If JMX requires authentication only (but no security), then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication.
Confirm Password	Confirm the Password by retyping it in this text box.
GemFire Servers	Specify the name of the server if the monitored GemFire server belongs to a cluster. By default, this is *none*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Total members	Indicates the total number of cache members associated with this region.	Number	
Average reads	Indicates the average number of read requests received by this region per second.	Reads/sec	
Average writes	Indicates the average number of write requests received by this region per second.	Writes/sec	
Bucket count	Indicates the total number of data buckets associated with this region of this region.	Number	Geode automatically determines the physical location of data in the members that host a partitioned region's data. Geode breaks partitioned region data into units of storage known as buckets and stores each bucket in a region host member. Buckets are distributed in accordance to the member's region attribute settings.
Number of bucket entries	Indicates the number of data buckets associated with this region to which	Number	When an entry is created, it is assigned to a bucket. Keys are grouped together in a bucket and always remain there. If

Measurement	Description	Measurement Unit	Interpretation
	data was put.		the configuration allows, the buckets may be moved between members to balance the load.
Create rates	Indicates the rate at which data was added to this region.	Creates/sec	
Destroy rates	Indicates the rate at which object entry was destroyed from this region.	Destroys/sec	
Disk read IOPS	Indicates the rate at which data was read from the disk associated with this region.	Reads/sec	Comparing the value of these measures across the regions helps administrators identify the region with the maximum disk read IOPS and disk write IOPS.
Disk write IOPS	Indicates the rate at which data was written to the disk associated with this region.	Writes/sec	
Disk IOPS	Indicates the rate at which I/O operations were performed on the disk associated with this region.	IOPS	
Disk usage	Indicates the amount of data utilized by the disk associated with this region.	MB	
Gets rate	Indicates the rate at which get operations were successful on this region.	Gets/sec	
Putall rates	Indicates the rate at which a map was added or replaced on this region as a result of a local operation.	Puts/sec	
Puts rate	Indicates the rate at which data was added or	Puts/sec	

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td></td><td>replaced on this region.</td><td></td><td></td></tr>
<tr><td>Is gateway enabled ?</td><td>Indicates whether/not this region sends data through a gateway.</td><td></td><td>The values reported by this measure and its numeric equivalents are mentioned in the table below:
<table><tr><th>Measure value</th><th>Numeric Value</th></tr><tr><td>Yes</td><td>1</td></tr><tr><td>No</td><td>0</td></tr></table>
Note:
By default, this measure reports the Measure Values listed in the table above to indicate whether/not this region sends data through a gateway. The graph of this measure however, represents the status of a server using the numeric equivalents only - 0 or 1.</td></tr>
</table>

4.1.4 GemFire Resource Usage Test

By closely monitoring the reboot status, CPU usage, throughput and the I/O activity of each member, and comparing these metrics across members, you can quickly isolate resource-intensive members.

Target of the test : A GemFire Cluster server

Agent deploying the test : An internal/remote agent.

Outputs of the test : One set of results for each resource:member connected to the target GemFire server being monitored.

Configurable parameters for the test

Parameters	Description
Test Period	How often should the test be executed.
Host	The host for which the test is to be configured.

Parameters	Description
Port	The port on which the specified host listens. By default, this is 7070.
JMXPort	Here, specify the port at which the jmx listens for requests from remote hosts. Ensure that you specify the same port that you configured in the management.properties file in the <JAVA_HOME>\jre\lib\management folder used by the target application
JNDIName	The JNDIname is a lookup name for connecting to the JMX connector. By default, this is jmxrmi. If you have resgistered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
JMX UserName and JMX Password	If JMX requires authentication only (but no security), then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication.
Confirm Password	Confirm the Password by retyping it in this text box.
GemFire Servers	Specify the name of the server if the monitored GemFire server belongs to a cluster. By default, this is *none*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Is running?	Indicates whether/not this member is currently running.		The values reported by this measure and its numeric equivalents are mentioned in the table below: **Measure value** / **Numeric Value** No / 0 Yes / 1 Unavailable / 2 **Note:** By default, this measure reports the **Measure Value**s listed in the table above to indicate whether/not the member is currently running.. The graph of this measure however, represents the status of a server using the numeric equivalents only - 0 or 1.

<table>
<tr><th>Measurement</th><th>Description</th><th>Measurement Unit</th><th>Interpretation</th></tr>
<tr><td>Is rebooted?</td><td>Indicates whether/not this member is rebooted.</td><td></td><td>The values reported by this measure and its numeric equivalents are mentioned in the table below:
<table><tr><th>Measure value</th><th>Numeric Value</th></tr><tr><td>Yes</td><td>0</td></tr><tr><td>No</td><td>1</td></tr></table>
Note:
By default, this measure reports the Measure Values listed in the table above to indicate whether/not the member is rebooted. The graph of this measure however, represents the status of a server using the numeric equivalents only - 0 or 1.</td></tr>
<tr><td>Active executions</td><td>Indicates the number of executions i.e., Continuous queries (CQs) that were active for this member.</td><td>Number</td><td></td></tr>
<tr><td>CPU usage</td><td>Indicates the percentage of CPU utilized by this member.</td><td>Percentage</td><td></td></tr>
<tr><td>Host CPU usage</td><td>Indicates the percentage of time for which the CPU of this member's host was in use.</td><td>Percentage</td><td></td></tr>
<tr><td>Uptime</td><td>Indicates the total amount of time for which this member was up and running.</td><td>Seconds</td><td></td></tr>
<tr><td>Total regions</td><td>Indicates the total number of regions present in the cache allocated to this member.</td><td>Number</td><td></td></tr>
</table>

Measurement	Description	Measurement Unit	Interpretation
Average reads	Indicates the average rate at which data was read from the cache of this member.	Reads/sec	Compare the value of these measures across the members to figure out the most resource intensive member.
Average writes	Indicates the average rate at which data was written to the cache of this member.	Writes/sec	
Data sent	Indicates the amount of data sent from the cache of this member.	MB/sec	Compare the values of these measures across the members to figure out the member through which maximum amount of data is transmitted/received.
Data received	Indicates the amount of data received by the cache of this member.	MB/sec	
Clients connected	Indicates the number of client connections to this member.	Number	
Total locator	Indicates the total number of locators in the GemFire server.	Number	This measure is applicable only for the 'Summary' descriptor.
Total server	Indicates the total number of backend servers communicating with the GemFire server.	Number	This measure is applicable only for the 'Summary' descriptor.

4.1.5 GemFire Workload Test

Periodically, administrators should measure the workload on the GemFire Cluster server and evaluate the server's ability to handle the load, so that they can figure out whether the server's size commensurate to its load or not. Administrators can perform this load analysis using the **GemFire Workload** test.

This test tracks the load on the GemFire Cluster server and reports the number threads currently catering to the user requests. In addition, this test reports the time taken for each thread to cater to the user requests as well as the connections that failed. Using this test administrators can figure out

the count of threads that are generating the load on the GemFire Cluster server. Capacity planning and clustering decisions can be taken based on insights provided by this test.

Target of the test : A GemFire Cluster

Agent deploying the test : An internal/remote agent.

Outputs of the test : One set of results for each server/application that is connected to the target GemFire Cluster being monitored.

Configurable parameters for the test

Parameters	Description
Test Period	How often should the test be executed.
Host	The host for which the test is to be configured.
Port	The port on which the specified host listens. By default, this is 7070.
JMXPort	Here, specify the port at which the jmx listens for requests from remote hosts. Ensure that you specify the same port that you configured in the management.properties file in the <JAVA_HOME>\jre\lib\management folder used by the target application
JNDIName	The JNDIname is a lookup name for connecting to the JMX connector. By default, this is jmxrmi. If you have resgistered the JMX connector in the RMI registery using a different lookup name, then you can change this default value to reflect the same.
JMX UserName and JMX Password	If JMX requires authentication only (but no security), then ensure that the user and password parameters are configured with the credentials of a user with read-write access to JMX. To know how to create this user, refer to Configuring the eG Agent to Support JMX Authentication.
Confirm Password	Confirm the Password by retyping it in this text box.
GemFire Servers	Specify the name of the server if the monitored GemFire server belongs to a cluster. By default, this is *none*.

Measurements made by the test

Measurement	Description	Measurement Unit	Interpretation
Maximum connections	Indicates the maximum number of connections allowed to the GemFire server.	Number	

Measurement	Description	Measurement Unit	Interpretation
Maximum threads	Indicates the maximum number of threads allowed to service user requests.	Number	
Ping time	Indicates the maximum time taken by the server to cater to the user requests.	Milliseconds	A low value is desired for this measure.
Active executions	Indicates the number of executions i.e., CQs (Continuous Queries) that were active on the GemFire server.	Number	
Average latency for processing client notifications	Indicates the average time taken by the server to process user requests.	Milliseconds	A low value is desired for this measure.
Client connection threads	Indicates the number of threads that are currently catering to the user requests.	Number	
Client virtual machines connected	Indicates the number of virtual machines that are connected to the GemFire server in a cluster setup.	Number	
Get average latency	Indicates the average time taken to retrieve data from the backend servers by the GemFire server to cater to the user requests.	Millseconds	
Get request rate	Indicates the rate at which data was retrieved from the backend server by the GemFire server upon user requests.	Requests/sec	
Put average latency	Indicates the average time taken by the GemFire server to send the data to the backend servers.	Milliseconds	

Measurement	Description	Measurement Unit	Interpretation
Put request rate	Indicates the rate at which data is sent to the backend servers from the GemFire server.	Requests/sec	
Number of connections waiting for a thread	Indicates the number of connections that are currently waiting for a thread to start processing the user request.	Number	A low value is desired for this measure.
Failed client connections	Indicates the number of connections that failed to connect to the GemFire server.	Number	Ideally, the value of this measure should be zero.