Proven Practice

**Tuning Cognos ReportNet for a High Performance Environment**

Product(s): Cognos ReportNet
Area of Interest: Performance
Copyright

Your use of this document is subject to the Terms of Use governing the Cognos software products and related services which you have licensed or purchased from Cognos. The information contained in this document is proprietary information of Cognos Incorporated and/or its licensors and is protected under copyright and other applicable laws. You may use the information and methodologies described in this document 'as is' or you may modify them, however Cognos will not be responsible for any deficiencies or errors that result from modifications which you make. Copyright 2006 (c) Cognos Incorporated. All Rights Reserved.

You can print selected pages, a section, or the whole book. Cognos grants you a non-exclusive, non-transferable license to use, copy, and reproduce the copyright materials, in printed or electronic format, solely for the purpose of providing internal training on, operating, and maintaining the Cognos software.

This document is maintained by the Best Practices, Product and Technology team. You can send comments, suggestions, and additions to BestPractices-ProductandTechnology@cognos.com.
# Contents

1. **INTRODUCTION** .................................................................................................................. 4
  1.1 **PURPOSE** .................................................................................................................. 4
  1.2 **SCOPE** ..................................................................................................................... 4
  1.3 **OVERVIEW** ............................................................................................................... 4

2. **WHAT IS A REPORT PROCESS?** ....................................................................................... 5

3. **WHAT IS AN AFFINITY CONNECTION?** ......................................................................... 6
  3.1 **ABSOLUTE AFFINITY** ............................................................................................... 7
  3.2 **LOW AFFINITY** ......................................................................................................... 7
  3.3 **HIGH AFFINITY** ........................................................................................................ 8

4. **PDF CASE STUDY** .............................................................................................................. 9
  4.1 **SETTINGS FOR THE INTERACTIVE REPORT SERVICE:** ............................................. 9
  4.2 **SETTINGS FOR THE BATCH REPORT SERVICE:** ......................................................... 9
  4.3 **SETTINGS FOR THE MAXIMUM NUMBER OF JOBS:** ................................................. 10
  4.4 **SETTING THE QUEUE TIME LIMIT OF REPORT SERVICE IN SECONDS:** ................. 10
  4.5 **EXAMPLE OF RUNNING A PDF REPORT** .................................................................. 12

5. **HTML CASE STUDY** ......................................................................................................... 13
  5.1 **SETTINGS FOR THE INTERACTIVE REPORT SERVICE:** ............................................. 14
  5.2 **EXAMPLE OF RUNNING AN HTML REPORT** .............................................................. 15

6. **PROCESSING CAPACITY** ................................................................................................... 16

7. **TOMCAT APPLICATION SERVER TUNING PARAMETERS** .............................................. 16

8. **CHANGING THE DEFAULT FONT** .................................................................................... 18

9. **WEB SERVER TUNING** .................................................................................................... 18

10. **SECURITY AND AUDITING** ........................................................................................ 19

11. **QUI C K G U I D E** .......................................................................................................... 20
1 Introduction

1.1 Purpose

This document is a how-to guide on tuning and optimizing Cognos ReportNet from a system perspective. Application tuning such as report layout and query optimization will not be discussed. The document consists of detailed instructions, screenshots and examples that explain the steps required to tune ReportNet in a single server or distributed server environment. Some users may find this approach over simplified. It is important to note, that there has been a huge number of Cognos ReportNet users with a great misunderstanding on how the settings are calculated and applied. The primary goal is to have a solid understanding of the relationship between ReportNet report processes and affinity connections in order to optimize the use of available hardware resources in a production environment.

1.2 Scope

The instructions in this document will work across all platforms however the examples shown are based on a Windows install. Additional information can be found in the Architecture and Planning Guide and the Administration and Security Guide that ships with Cognos ReportNet. This document only applies to Cognos ReportNet 1.1 MR1 and 1.1 MR2. However many of the concepts in this document also apply to Cognos 8.

1.3 Overview

In Cognos ReportNet a request to process a report can either be initiated by users (i.e. interactive) or by events such as a schedule (i.e. batch). Cognos ReportNet interactive report processing is handled by the Report Service, while scheduled reports are handled by the Batch Report Service.

Given the flexible, service oriented architecture (SOA) provided by ReportNet, a Cognos topology can be configured in a variety of ways to best make use of available hardware resources. If interactive and batch processing periods are expected to conflict, the respective settings for each of the Report and Batch Report Service should be considered in tandem.
This document is intended to describe the ReportNet process/thread model used for report processing and offer guidance on tuning a system for optimal resource utilization. The ReportNet process/thread model relates to the settings configured for the Report Service and Batch Report Service.

For the purpose of discussion, we will consider the tuning parameters for the Report Service (invoked by user-driven report processing). A report process equates to a BIBusTKServerMain thread.

Note: When configuring ReportNet, you should always consider the tuning settings for the Report Service in relation to available hardware capacity. Hardware capacity is dictated by various factors including servers in a topology, number of CPUs, CPU clock rate, etc.

2 What is a Report Process?

The role of a report process is to execute activities related to report processing. Report processing activities include parsing meta-data for report specification information, generating and submitting SQL statements to a reporting data source and rendering data into a requested report output format (such as PDF).

A report process is identified as “BIBusTKServerMain” in Task Manager or in the Unix process list. The default setting is 2 report processes for every Report Server component installed in a given environment. Each process independently manages its own memory space and acts as a container for a configured number of High and Low affinity connections.

RECOMMENDATION: As a starting point the recommendation is to configure 2 report processes per physical CPU.

NOTE: The correct number of report processes will vary depending on individual client BI application.
Example configuration:

```
Server A – 2 CPUs @ 3.06 GHz with 4 GB memory

<table>
<thead>
<tr>
<th>CRN Report Process #1</th>
<th>CRN Report Process #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRN Report Process #3</td>
<td>CRN Report Process #4</td>
</tr>
</tbody>
</table>
```

Note: Maximum number of interactive report processes is set to 4

### 3 What is an Affinity Connection?

An affinity connection is used for request assignment purposes and is an internal working part of a CRN report process (BIBusTKServerMain). Affinity refers to whether a request is assigned to a specific server or, a load-balancing mechanism can assign it to another server. Affinity between request and server ensures that requests are routed to an appropriate computer for execution. There are three types of affinity: absolute, high, and low. For more information on affinity connection types, please refer to sections 3.1, 3.2 and 3.3.

**RECOMMENDATIONS:**

DO NOT attempt to change the affinity connection settings unless you thoroughly understand them because they can severely impact Cognos stability and/or performance on the system. If in doubt, contact Cognos Customer Support.

The default settings for Report Service affinity connections are usually adequate to handle most types of Cognos BI applications (High = 1, Low = 4).

The maximum number of total affinity connections per report process should NOT exceed 5 (i.e. sum of Low and High affinity connections).
NOTE: A request will be assigned to an affinity connection for the duration of the request (i.e. from request submission to request completion)

3.1 Absolute Affinity

In addition to low and high affinity connections per report process, there are also absolute affinity requests. These are requests that can only be executed by a specific report server, regardless of the load balancing used. The cancel report execution is a great example of this, you can only cancel a report on the server running it.

By its very nature, absolute affinity requests are just that – absolute, therefore tuning parameters for this type of request are not exposed in ReportNet to avoid redundancy.

Absolute affinity is used to create an association between the client and the executing server to ensure that long-running requests do not time out. An absolute affinity request is used with the following operations: wait, getOutput, and release.

Example: When a user cancels a running report, absolute affinity routes the cancel request back to the originating report process used to execute the report.

3.2 Low Affinity

A low affinity request is one that can be completed with the same efficiency by any report process. Low affinity requests are considered independent without any relationship to other requests that were processed by the system. On-demand reports which includes PDF, and the first page of an HTML report are good examples of low affinity requests.

*Reporting:* Report querying, Report processing

*Report Authoring:* Metadata retrieval, Query Validation

*Administrative:* Testing data source connections, Adding objects (folders, jobs, schedules, etc.), Refreshing portal page
Example 1: When a user runs an HTML report, any available report process (BiBusTKServerMain) can be used to render the first page. The process will use a low affinity connection for request assignment purposes. All other pages for that report will be assigned to a high affinity connection of the same report process if available. Otherwise if the high affinity connection of the same report process is not available it will relegate processing to a Low affinity connection.

Example 2: When a user runs a PDF report, any available report process (BiBusTKServerMain) can be used to render the report. The process will use a low affinity connection for request assignment purposes to render the entire PDF report.

3.3 High Affinity

A high affinity requests is one that can be executed on any report process but would be most efficiently executed on a specific report process - page down on an interactive report is a great example of this. If the same report process is used, we scroll the existing database cursor, retrieve the data, and render the page.

In other words when the first page is rendered, information is cached by the report process so that subsequent High affinity requests can avoid overhead activities.
These activities include, preparing the SQL, parsing report specification needed to build the layout, etc.
This is accessed from cached information (i.e. report process memory).

If that process is not available, because all high affinity connections are assigned or the administrator shut down the computer or there was a network failure, the request is routed to another available report process for execution. The next page can still be served up, although execution will be slower.

NOTE: ReportNet routes high affinity requests to a specific server regardless of the load balancing used.

Report Viewer links: Run again, Return
HTML Report Navigation: Top page, Page up Page down Bottom page

Delivery Options: Save, Save As, Print, Email, Viewing

Example: When a user runs an HTML report, a BiBusTKserverMain process is spawned. That process will use a Low affinity connection to render the first page. When the user clicks on Page Down or Bottom, the High affinity connection will render all other pages (next pages and last). In other words the High affinity connection will render all pages except the First.

4 PDF Case Study

Scenario: A client has a 4 CPU server with 8G of RAM, and 80% of their reports will be in PDF format.

4.1 Settings for the Interactive Report Service:

As a starting point, the maximum number of interactive report processes should be set to 8 (2 report processes per CPU * 4 CPUs) as a starting point.

Since most of the reports will be in PDF format and knowing that PDF formatted reports are rendered using a low affinity connection, the default settings of low and high affinity connections which are set to 4 and 1 respectively.

4.2 Settings for the Batch Report Service:

Since batch reports of any format are rendered by the low affinity connection, it is the only setting that requires consideration. Since these types of reports are usually long running and may be memory intensive, it may be beneficial to reduce the number low affinity connections from the default of 4. For a 32 bit application we know that each process is capable of using up to 2G of memory. This will allow each assigned affinity request to have a greater proportion of the addressable memory space by each report process.

As a starting point, the maximum number of batch report processes should be set to 8 (2 report processes per CPU * 4 CPUs). This is the same as the Interactive Report Service. Keep in mind that if Batch reports will run during interactive usage periods, the Maximum number of processes for the Report and Batch report service should be considered in tandem. The sum of batch and interactive report processes that can be spawned simultaneously should not exceed the available capacity of the server.
4.3 Settings for the Maximum number of Jobs:

The “Maximum jobs during peak period” is set based on what the clients business needs but it should not exceed the “Maximum jobs during non-peak period”. This setting is designed to limit the amount of batch processing activity while users are on the system.

The “Maximum jobs during non-peak period” is set based on the \((\text{Number of Low affinity connections for the Batch report service} \times \text{Max number of Batch report service processes})\). In this case it works out to\((2 \times 8 = 16)\)

4.4 Setting the Queue time limit of report service in seconds:

The queue limit refers to the maximum length of time that a submitted request has to start processing before an exception occurs. For the Report Service, this setting is dependent on the number of reports that need to run, the number of available processes to serve those reports and the length of time required to service a report in general. Environments with many requests and longer running reports might require this value to be increased.

Note - The default is 30 seconds. If this threshold is exceeded it will result in the following error:
DPR-ERR-2002 Unable to execute the request because there was no process available within the configured time limit. The server may be busy.

The value set should coincide with the particular client’s performance expectations. For more guidance on what value to enter contact Cognos Customer Support.

Results: Based on all the settings mentioned for the PDF case study this is what the Server Configuration will look like
The above settings are a baseline. Additional time should be spent monitoring the system to determine whether adjustments to the settings are required for optimal system utilization.

Latency is the length of time that a submitted request is queued in the system. If in the Latency column, it is always showing 0/xx (xx refers to the queue time limit) then that means that nothing is being queued. This is an indicator that all requests are being assigned in a timely manner.
If the Latency was showing x/xx constantly, and the server is NOT at 100% CPU utilization then that would be an indicator that the system is able to accommodate more requests. This is accomplished by increasing the maximum number of interactive report processes and monitor again. As an example increasing by 2 would bring the above settings from 8 to 10.

### 4.5 Example of running a PDF report.

| Number of High affinity connections for the interactive report service | 1 |
| Number of low affinity connections for the interactive report service | 4 |
| Maximum number of interactive report service processes | 0 |

Based on the settings above, when a user runs a PDF report, Report Process #1 (out of 8) will be spawned and a Low affinity connection will be “Assigned” to serve up the report.
Once the report request has been completed the Low affinity connection will be “Not Assigned” once again.

5 HTML Case Study

Scenario: A client has a 4 CPU server with 8G of RAM, and 80% of their reports will be in HTML format.
5.1 Settings for the Interactive Report Service:

As a starting point, the maximum number of interactive report processes should be set to 8 (2 report processes per CPU * 4 CPUs) as a starting point.

Since most of the reports will be in HTML format and knowing that all pages other than the first page are rendered by the High affinity connection, consideration should be given to the appropriate mix of High and Low affinity connections. As an example, for this case study we could set Low affinity connections to 3 and High affinity to 2.

Additional monitoring will be required to see if the mix of High and Low affinity connections is correct.

The settings for the Batch report service and maximum number of Jobs do not change. They will be the same as outlined in the PDF case study.

NOTE:
The settings stated in the examples above are a starting point and are not meant to be set once and forgotten. It is an evolving process that through monitoring and adjustment will lead to obtaining the optimum settings (sweet spot) for a given server.
### 5.2 Example of running an HTML report

<table>
<thead>
<tr>
<th>Tuning</th>
<th>Number of high affinity connections for the interactive report service</th>
<th>Interactive Report Service</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tuning</th>
<th>Number of low affinity connections for the interactive report service</th>
<th>Interactive Report Service</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tuning</th>
<th>Maximum number of interactive report service processes</th>
<th>Interactive Report Service</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>No</td>
</tr>
</tbody>
</table>

Based on the settings above, when a user runs an HTML report, Report Process #1 (out of 8) will be spawned and a Low affinity connection will be assigned to serve up the first page of the report. When the user clicks on “page down”, this type of request will be served up by the High affinity connections.

Once the request has completed the first page of the report, the Low affinity connection will be “Not Assigned” once again. When a page down request is submitted, it will be assigned to a High affinity connection if available.
6 Processing Capacity

If in a ReportNet environment there is more than one dispatcher, you can specify the proportion of requests that each dispatcher handles via the processing capacity setting. The capacity for a dispatcher should be set based on the number of CPUs and the CPU speed in comparison to other dispatchers in a given environment.

Example:
Assuming all other factors are equal, a first dispatcher is installed on a 2 CPU computer and a second dispatcher on a 4 CPU computer. The processing capacity of the first dispatcher is set to 1.0 and the second to 2.0. The second dispatcher handles two-thirds of the requests while the second handles one-third of the requests. If the capacity of both dispatchers is set to 1.0, requests are sent to each dispatcher in an equal fashion.

The default processing capacity for each dispatcher is 1.0

7 Tomcat Application Server Tuning Parameters

The two main considerations for supported J2EE application server tuning are adjusting the JVM heap size (i.e. memory), and the required number of threads to accommodate expected user load.

RECOMMENDATION:
Memory – 1024M per JVM instance

Threads – at least 2.5 times the number of expected concurrent users

The following settings are specifically for Apache Tomcat Application server only which ships with Cognos ReportNet.

Memory:
Apache Tomcat memory settings can be configured through the Cognos Configuration UI (3 configuration choices - small, medium and large) or through the startup.bat file residing in <installLocation>\crn\bin

If Cognos ReportNet is started as a service, then the Cognos Configuration settings will apply.
If Cognos ReportNet is started via the startup.bat file, the custom settings within the file will apply
Both the configuration tool and the `startup.bat` file accomplish the same thing. The Cognos Configuration tool sets it for the service, whereas the `startup.bat` file sets it for manual start via a batch file.

In either case, the memory settings are used to set the `java.exe` process (JVM size), which relates to the memory available for use by the configured Cognos ReportNet Java based services. This includes Cognos ReportNet dispatchers, presentation service, Job and Scheduling service and Content Manager.

A good starting point for servers with greater than 3G of memory is to set the JVM size to 1024.

Threads:
Adjust the “coyote” settings in the `server.xml` file residing in
`<installLocation>\crn\tomcat4.x.xx\conf.`

Cognos internal testing has indicated that setting the `maxProcessors` setting to 1000 connections instead of the default 75 may be advantageous to performance and scalability. This will determine the number of available threads that can be handled by the Apache Tomcat process. Since Tomcat has a smaller footprint than other application servers setting this value to 1000 is not detrimental.

Also, change the `acceptCount` from 100 to 500.

Before:

```xml
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector" port="9300" minProcessors="5" maxProcessors="75" enableLookups="true" redirectPort="9443" acceptCount="100" debug="0" connectionTimeout="60000" useURIValidationHack="false" disableUploadTimeout="true"/>
```

After:

```xml
<!-- Define a non-SSL Coyote HTTP/1.1 Connector on port 8080 -->
<Connector className="org.apache.coyote.tomcat4.CoyoteConnector" port="9300" minProcessors="5" maxProcessors="1000" enableLookups="true" redirectPort="9443" acceptCount="500" debug="0" connectionTimeout="60000" useURIValidationHack="false" disableUploadTimeout="true"/>
```
8 Changing the default Font

Cognos ReportNet ships with the Andale WT font embedded in the product in order to ensure that the product can render all supported languages. The drawback is that the Andale font is 20M size. There is an additional cost for each report processed as well as, performance trade off.

To resolve this select the lightest font that can support the (client’s) application language requirements.

The Arial font is only 100Kb in size

9 Web Server Tuning

There are specific tuning parameters that each Web Server vendor has and which this document is not intended to cover any 3rd party settings.

The recommend use of scalable gateway implementations is:

IBM/Apache HTTP Server - Apache Mod

IIS - ISAPI
Netscape/iPlanet Web servers – servlets
Application Servers – servlets

10 Security and Auditing

Security and Auditing have great performance relevance.

Security
The use of SSL, Firewalls, and database security for example has a performance impact because CRN has to encrypt, decrypt, authenticate etc.
In other words, more stringent authorization means more CAM resources.

Auditing
The higher the Auditing level the more logging resources are used.

With Level 1 being least expensive and Level 5 being most expensive, the performance tradeoff for the different levels of auditing is:

Cost from 1 to 3 rises linearly
Cost from 3 to 4 remains fairly constant
Cost from 4 to 5 rises dramatically (this is DEBUG mode)
## 11 Quick Guide

<table>
<thead>
<tr>
<th>Component/Service</th>
<th>Setting</th>
<th>Default Value</th>
<th>Recommended Starting Value</th>
<th>Reason/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReportNet Dispatcher</td>
<td>Process Capacity</td>
<td>1.0</td>
<td>For each server:</td>
<td>This setting dictates the number of BI requests that will be handled by a particular server in the ReportNet topology.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Appropriate value is a function of the number of CPUs, CPU clock rate, memory, etc in relation to other servers in the topology.</td>
<td>By default, requests will be distributed so that each server is expected to handle an equivalent number of requests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>While it may be difficult to estimate a precisely correct set of values for heterogeneous server environments, there are several methods that can improve the relative accuracy of this task.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Use the relative performance ratings of a server provided by hardware vendors. Estimate the process capacity for a server using one variable at a time in the order of (# of CPUs, clock rate of CPUs)</td>
</tr>
<tr>
<td>Component/Service</td>
<td>Setting</td>
<td>Default Value</td>
<td>Recommended Starting Value</td>
<td>Reason/Comments</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>---------------</td>
<td>-----------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>ReportNet Dispatcher</td>
<td>Process Capacity</td>
<td>1.0</td>
<td>For each server: 1.0 Appropriate value is a function of the number of CPUs, CPU clock rate, memory, etc in relation to other servers in the topology.</td>
<td>Monitor the request rate through the Server Administration tool of Cognos Connection. Isolate the Report services to monitor interactive usage. This will indicate whether the applied process capacity settings lead to expected behavior. Key metrics to observe include Latency and Seconds per request. If latency is building up on one report service while others are functioning freely, reduce the process capacity on the high latency server.</td>
</tr>
<tr>
<td>Report</td>
<td>Maximum report service processes</td>
<td>2</td>
<td>For each server: 2 * # of physical CPUs.</td>
<td>This setting dictates the number of report processes (BiBusTKserverMain) that will be spawned to handle interactive processing activity. Must be considered in combination with Batch Report service settings.</td>
</tr>
<tr>
<td></td>
<td>Number of high affinity connections (per report process)</td>
<td>1</td>
<td>For each report process: 1</td>
<td>This setting indicates the number of threads available per interactive report server (report process) to handle high affinity requests. This setting must be considered along with the low affinity connections setting.</td>
</tr>
<tr>
<td>Component/Service</td>
<td>Setting</td>
<td>Default Value</td>
<td>Recommended Starting Value</td>
<td>Reason/Comments</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>---------------</td>
<td>----------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>Number of low affinity connections (per report process)</td>
<td>4</td>
<td>For each report process: 2 to 4</td>
<td>This setting indicates the number of threads available per interactive report server (report process) to handle low affinity requests. This setting must be considered along with the high affinity connections setting.</td>
</tr>
<tr>
<td></td>
<td>Number of low + high affinity connections</td>
<td>5</td>
<td>Max = 5</td>
<td>Regardless of the combination selected, the number of high and low affinity connections should NOT exceed 5 per report process. Managing a large number of threads is costly from an overhead perspective and a single report process has a memory threshold ceiling.</td>
</tr>
<tr>
<td>Report cont’d</td>
<td>Queue timeout limit</td>
<td>30 sec</td>
<td>300 sec</td>
<td>This setting is used to determine the length of time in seconds in which a request will be queued in a ReportNet dispatcher before being terminated. A request will reside with the dispatcher until an appropriate high/low affinity connection becomes available.</td>
</tr>
<tr>
<td>Component/ Service</td>
<td>Setting</td>
<td>Default Value</td>
<td>Recommended Starting Value</td>
<td>Reason/Comments</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
<td>---------------</td>
<td>----------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Batch Report</td>
<td>Maximum batch report service processes</td>
<td>2</td>
<td>For each server: 2 * # of CPUs</td>
<td>This setting dictates the number of report processes (BiBusTKserverMain) that will be spawned to handle scheduled processing activity. Must be considered in conjunction with Report service settings. Overlap in interactive and scheduled activity may warrant a lowering of this threshold to accommodate both processing activities.</td>
</tr>
<tr>
<td></td>
<td>Number of high affinity connections (per batch report process)</td>
<td>1</td>
<td>For each batch report process: 0</td>
<td>This setting is not applicable to scheduled activity processing. Current scheduling functionality does not possess any high affinity requests</td>
</tr>
<tr>
<td></td>
<td>Number of low affinity connections (per batch report process)</td>
<td>4</td>
<td>For each batch report process: 2</td>
<td>This setting indicates the number of threads available per batch report server (report process) to handle low affinity requests. This setting must be considered along with the high affinity connections setting.</td>
</tr>
<tr>
<td>Component/ Service</td>
<td>Setting</td>
<td>Default Value</td>
<td>Recommended Starting Value</td>
<td>Reason/Comments</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------</td>
<td>---------------</td>
<td>----------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Job and Scheduling Service</td>
<td>Peak period start</td>
<td>7</td>
<td>No specific value recommended</td>
<td>Dependent on daily peak periods defined by application usage.</td>
</tr>
<tr>
<td></td>
<td>Peak period end</td>
<td>18</td>
<td>No specific value recommended</td>
<td>Dependent on daily peak periods defined by application usage.</td>
</tr>
<tr>
<td></td>
<td>Maximum number of jobs during peak</td>
<td>1</td>
<td>Set to a value less than Maximum number of jobs during non-peak setting whereby desired allocation between interactive and batch report services are processed</td>
<td>If using jobs or schedules during interactive usage periods, this setting will limit the number of jobs that can be executed and that will contend for resources with report services.</td>
</tr>
<tr>
<td></td>
<td>Maximum number of jobs during non-peak</td>
<td>4</td>
<td>For all servers: {Maximum batch report service processes * Number of low affinity threads}</td>
<td>If using jobs or schedules to drive batch reporting requirements, this setting will limit the number of jobs that can be executed simultaneously.</td>
</tr>
</tbody>
</table>