IBM Tivoli Netcool/Impact 7.1
Sizing and Tuning Guide

Author: Jeffrey D. Jones
Software Engineer

Oct 2014

CONTENTS

LIST OF TABLES

REVISION HISTORY

Date	Version	Revised By	Comments
07/25/2014	1.0	Jeff D. Jones	Creation
08/06/2014	1.1	Jeff D. Jones	Review updates
09/05/2014	1.2	Jeff D. Jones	Review updates
09/26/2014	1.3	Jeff D. Jones	Review updates
10/20/2014	1.4	Jeff D. Jones	Review updates

1 Introduction

This document is an overview of some of the considerations to be made when sizing and tuning your Netcool/Impact systems. The objective is to provide the sizing, tuning or product software configuration recommendations to achieve the proper product performance for the event capacities occurring in your environment. This document is not meant to serve as a comprehensive or formal guide, but merely provide a good place to start when initializing your Impact sizing requests.

These recommendations are based upon performance observation during the product Performance Verification Test. Tuning the Netcool/Impact Service start with identifying the current Impact JVM settings and enabling JVM Garbage Collection (GC) logging. Next, an event representative workload was run to determine Impact system size boundaries. The size boundaries were determined by analyzing the system GC logs, CPU, disk, Memory and network statistics.

2 Sizing your Netcool/Impact System

The most important consideration when sizing your Netcool/Impact system resource requirements will be the size of the average and burst event rates occurring in your environment. These event estimates will determine your Netcool/Impact system CPU, memory and disk requirements. Your initial sizing estimates should be based upon these event metrics and then adjusted to compensate for other metrics as needed.

The sizing for this release started with the sizing estimates for the previous product, IBM Tivoli Netcool/Impact 6.1.1 as a guide and made adjustments as necessary based on the performance results. As a baseline, the sizing capacity of each system was determined at the point where the CPU utilization reached 40 percent at the maximum event processing throughput. Based on the current performance data, the Netcool/Impact 7.1 sizing recommendations fall into small, medium and Large Netcool/Impact systems categories.

The following system sizing examples are based on a single simple enrichment policy retrieving one row from a DB2 database for enrichment. However if executing multiple polices or services your event processing throughput could be different. A small system is recommended mostly for test or to verify system functionality. Use the following link for additional sizing requirements.

Software Product Compatibility Reports

This sizing example produced the following sizing recommendations; however, in a production environment it is recommended to have at least a two server clustered system for failover capability.

2.1 Small System

In a small system, the event capacity will be 250 events per second or less. The deployment will be a single impact server consisting of the Impact GUI and backend server. Although the minimum specification is generally not to be recommended, there are some customers with environments where the event rates are very low for which the minimum specification may be appropriate. The system resource estimates for a small system are the following:

- 2 Core Processor
- 4GB RAM (sufficient in these cases)

2.2 Medium System

Most systems sizing requests will tend to fall into this general category. A medium system is a single Impact server with an event capacity of 250 to 350 events per second. The recommended system resources are the following:

- 4 Core Processor
- 8GB RAM

2.3 Large System

A large system consists of two or more servers in a cluster. The normal event rate in this size category will be between 350 to 500 events per second. This size system occasionally will experience momentary burst or steady state event rates above 500 events per second. The user in this sizing category requires failover capability with the ability to maintain the normal stated event processing throughput load in case a server goes offline or fails.

Your maximum event rate processing throughput for this size system will vary depending on network, event rate, event enrichment requirements and data sources performance capacities. A Netcool/Impact two server system is capable of handling momentary event rates up to 1000 events per second. The following list shows the appropriate sizing request for each server:

- 6-8 Core Processor
- 12-16GB RAM

The resource requirements for each server in the cluster is the same as the single server. Most customers also deploy an additional GUI server to provide multiple access points to the Netcool/Impact cluster. This size of system will require some additional tuning requirements that are described in the tuning section.

2.4 CPU Recommendations

Tivoli product specifications for processors typically recommend a minimum of 3.0GHz processors; however, modern processor architectures are achieving higher rates of throughput by sacrificing pure speed for more cores. For applications such as Netcool/Impact that are highly multithreaded and benefit from parallelization, it’s possible to achieve the same performance using slower processor with more cores. In many cases, it may be entirely appropriate to deploy Netcool/Impact on a system with 4+ processing cores that are only 2.0GHz, or even slower.

2.5 Policy Considerations

It is important to consider the types of policies that are being executed on the Impact server. If the policies mostly involve executing data retrieval requests to external sources and event enrichment, the CPU is not likely to be overly taxed. However, if the policies that will be run involve significant data parsing or number crunching, they will be more CPU-intensive. In such cases, it is a good idea to add another processing core or two to the sizing recommendation, which will provide the Event Processor the additional resources to handle more CPU-intensive workloads.

2.5.1 Netcool/Impact JVM Heap Considerations

For memory intensive workloads, tuning the Impact Server JVM options is recommended. In this sizing example, the default JVM options were used with no additional adjustments. Policy executions involved mostly data retrievals for event enrichment.

The default JVM options are set as the following:

- Starting JVM Heap Size 512M (Xms)
- Maximum JVM Heap Size 1200M (Xmx)

For memory intensive workloads, adjust the starting and maximum JVM heap sizes according to your system memory capacity and event processing workload needs. Change your Netcool/Impact JVM options by editing the “jvm.options” file located in the $IMPACT_HOME/wlp/usr/servers/<instance-name> directory. The instance name will be the value you set for the Impact instance name during installation. A server restart is required after making these changes.

A typical update will be to double the starting and maximum JVM heap sizes for memory intensive workloads and then observe the JVM performance using a tool like the IBM Pattern Modeling and Analysis Tool for Java Garbage Collector, PMAT. The PMAT tool will give you an analysis as to whether there were any issues with the JVM heap size and give you recommendations for adjustments. The tool also allows you to graph various generation heap total memory, tenured and nursery memory measurements before and after GC. These graphs allow you to see how your Impact server is using JVM heap memory under event processing loads.

If you normally expect to experience memory intensive workloads, you would set the starting JVM heap size to match the observed JVM memory utilization above. Set the maximum JVM heap size to cover your expected event burst event workloads. See the link in the appendix section for information on obtaining the PMAT tool.

2.6 Impact GUI Server Sizing

The load on the GUI server connected solely to the Impact server is typically very low. If the GUI server is running on the same system as the Impact server, merely adding 2GB RAM to the determined requirement for the Impact server should be sufficient. For example, if it’s recommended that a particular Impact deployment should run on a system with 4 processing cores @ 2.0GHz and 8GB RAM, then the GUI server can be deployed on the same system if the total RAM is increased to 10GB.

Note here, when accessing multiple datatypes, enable them for UI Provider or when you have data type caching enabled, your GUI server memory consumption could be higher.

2.7 Clustered Deployment Considerations

Generally, customers who are interested in clustered Netcool/Impact deployments want the benefits of having a failover system or load balancing the event processing load across multiple Impact servers. Therefore, the cluster members should be sized no differently than single Impact server. For example, in a 2-server cluster environment each server should have the appropriate resource capacity for handling the normal event load in order to avoid downtime should one of the servers in the cluster go offline.

Typically, the largest cluster member, in other words the server with the largest amount of resources: memory, processor capacity or disk space should be the primary server. Typically the first server started in the cluster becomes the primary Impact server. For failback implementation, to designate your primary server, add the primary flag to the server properties file. The server properties file is located in the $IMPACT_HOME/etc directory. The server properties filename is “<instance-name>_server.props.” The server instance name is the value you set for the Impact instance name during installation, example “NCI_server.props.” Add the following line to the server properties file:

- impact.server.preferredprimary=true

The primary server in clustered environments will fetch events and process events. Secondary servers will only process events. Since the primary server runs the event readers/listeners and process events, the CPU utilization will be slightly higher on the primary Impact server.

See the Data Source Connection Pool and Event Processing Threads section for details on setting secondary clusters Event Processor Thread Manager parameters.

3 Tuning the Impact Server

This release as observed from the performance results is not CPU intensive. No additional JVM starting and maximum heap adjustments will be required to achieve your optimal event processing throughput using simple event enrichment. However, there are other software configurations to consider which will ensure optimal event processing throughput performance. Also for polices which does extensive data parsing as compared to event enrichment, you can expect higher CPU utilization.

3.1 Data Type Caching

If you frequently view data items using the UI or execute policies using the “GetByKey” or “GetByfilter” functions in your Netcool/Impact environment, then data type caching can reduce the total number of direct queries made to your data sources. Data type caching reduces the load on external data sources and increases system performance by allowing frequently used data to be stored in a cache. When you view data items in the UI or execute polices to retrieve data, the information is fetched from the cache rather than the database.

The Impact data type caching configuration allows you to determine what type of caching to enable and how much data to store and for how long. Caching is best suited for static data where the data is not changing often. When setting your cache size, note that a large cache size will use significantly more memory, but will save your time when the same data is referenced often. See the Netcool/Impact documentation link in appendix the section.

3.2 Event Reader, Event Processor and Data Source Connection Pool Tuning

During the product performance verification test cycle, it was observed that adjusting the default data source connection pool and event processing threads was necessary to maintain a balanced and stable event processing throughput execution. Table 1 and Table 2 on page 10 and 11 show the recommended settings for a system with a 6 core processor at 1.87 GHz and 16 GB RAM. The Object Server host used identical specifications to the Impact server in this sizing estimation. The

DB2 data source used for event enrichment data used a 4 core processor at 2.9 GHz with 8 GB of RAM.

3.2.1 Event Reader Tuning

The event reader has settings to control its polling interval, SQL query fetch size and the maximum queue size. You will make adjustments to these settings based of the event rate occurring in your environment. The polling interval adjustment can be done from the GUI or by editing the event reader properties file. The other settings are done in the properties file only.

The following table shows an event reader configuration used in this sizing example. Here the event reader is being set to poll the event source every two seconds for new or updated events. When a new event is detected, the event reader will then trigger your event enrichment policy. The event reader restriction filter determines which events are new and will be processed by the enrichment policy. Your event reader settings may vary depending on your event requirements and system capacities. However, these configurations will serve as a starting point for your tuning adjustments.

General Settings				Event Mapping	
Polling Interval (milliseconds)	**Startup Automatically when server starts**	**Service Log Write to file**	**Collect Reports Enable**	**Event Matching**	**Actions**
2000ms	unselected	selected	unselected	Test events with all filers	Get updated events

Table 1 Event Read Configuration

The event reader default “maxtoreadperquery” and “maxqueuesize” sizes are 1000 and 2000 events respectively. The queue size determines the maximum number of event records held in memory for event processing. The “maxtoreadperquery” setting determines the SQL record fetch size per query. The event reader will stop retrieving events when the maximum queue size is obtained. There is also a “queueSizeLowerLimit” when reached by the Event Processor processing the events from the EventReader EventQueue, the event reader will resume fetching events from the event source again. The “queueSizeLowerLimit” is calculated as 75% of the “maxqueuesize”, which is set at the default value of 1500. The default event reader fetch interval and fetch size is set at 3000ms and 1000 records.

When a high volume of events are occurring in your environment, some adjustments to the event reader fetch interval, fetch size and maximum queue parameters may be needed to provide the Event Processor with the correct events flow to keep up with the event rate in your environment.

Note here that if you experience momentary bursts or steady state event rates near or above your observed maximum event throughput processing capacity, changes to the event reader default settings may be necessary. The event processing throughput is directly related to the event reader polling interval and the number of records being read per the event reader SQL query of the event data source. At high event rates, adjustments need to be made to avoid over running your event reader.

Consider an event reader configured to poll the event source every 2 seconds while fetching 1000 records, the reader is fetching 500 events for processing every second on an average. If 600 events per second are occurring in your environment, the event rate is over running the event reader by 100 events per second. In this state, you will experience some delay in the average events being read and processed by Netcool/Impact.

For this overrun case, adjustments can be made to the event reader polling interval, the SQL fetch size and the maximum queue size. You will edit the event reader properties file located in the Netcool/Impact home “etc” directory. The file is called <instance_name>_<eventreadername>.props” where “instance_name” is your Impact server instance name and “eventreadername” is your event reader name in lower case. These adjustments must be performed for a single server and for all servers in your cluster.

In these overrun adjustments, the event reader is being adjusted for a faster polling interval and a larger SQL fetch and queue sizes. Add or edit the following lines in the event reader properties file as follows:

- impact.perftesteventreader.objectserver.maxtoreadperquery=2000
- impact.perftesteventreader.objectserver.polltime=2000
- mpact.perftesteventreader.maxqueuesize=4000

In this example my event reader was called “PerfTestEventReader.” You should use your event reader name instead (in lower case).

These settings allows the event reader to fetch 2000 events for processing every second, thus out pacing the 600 events rate occurring in the environment. The queue size is typically twice the size of the SQL fetch size. A restart of the Impact server is required.

3.2.2 Event Processor Service Threads Tuning

The Event Processor Service minimum and maximum thread setting controls the number of processing threads the service is allowed to handle the event load. The service uses a thread manager to set the number of active threads for your event load. The next table shows the event processor service configuration for the environment used in this sizing guide example.

Event Processor Service				
Minimum Number of Threads	**Maximum Number of Threads**	**Processing Throughput Maximize**	**Tuning Configuration Maintain on Restart**	**Service Log Write to file**
7	12	Selected	Unselected	Selected

Table 2 Event Processor Service Configuration

See the following Data Source Connection Pool and Event Processing Threads section for details on setting secondary clusters Event Processor Thread Manager parameters.

3.2.3 Data Source Connection Pool and Event Processor Service Thread Manager Tuning

The data source maximum connection setting determines the maximum number of JDBC connections Netcool/Impact will make to a data source to query or update the data source. The updates typically occur when executing policies which use SQL functions like BatchUpdate, DirectSQL or ReturnEvent. The number of connections are managed by the Impact server to optimize the event load. The server will automatically adjust the number of connections up to the maximum connection pool size in order to maintain an optimal event processing throughput for the event load.

The connection pool should be set to maintain the optimal event processing throughput for your environment. For your data source, you should set the connection pool size to handle the maximum event per second rate load that you are expecting in your environment. For example, you can determine this level by observing the number of events that are being enriched per second and make adjustments as necessary. This can be done by querying the event source to determine the number of unprocessed events. You can do this by adding an event processed flag to the object server "alerts.status" table. Your event enrichment policy would then update to flag when the event enrichment is complete. This total will give you an indication of the number of events that are being processed and if any adjustments are necessary.

Note here, you must also factor in the additional load on your database subsystem by the Impact server when increasing the connection pool size. This is because if your database server is overtaxed, it could have a negative performance affect.

The Event Processor Service Thread Manager will adjust the total number of threads continuously as necessary to meet the demands of the event workload. The thread manager will start with the minimum number of Event Processor threads during server startup. But, if you have enabled the "**Tuning Configuration – Maintain on Restart"** option for the Event Processor service, the Thread Manager will start with the number of threads the Event Processor service was using before its previous shutdown. This can be any number between the Minimum and Maximum number of Event Processor threads. Consider your minimum expected event rate when making these adjustments.

If your event rate is somewhat constant, you can set the minimum and maximum thread size setting to be the same value. This will turn off the Event Processor Thread Manager and maintain a constant number of threads to handle the event load. Also note that there will be a level of processing threads for which an event data source performance will decline or deteriorate. This will be level where the data source is being overtaxed.

The maximum data source connection pool should be equal to or greater than the maximum event processor threads. This will prevent the Event Processor Service having to wait for a data source connection.

The following table shows the data source connection pool sizing for DB2 and the Object Server for small to larger environment using the system configurations in this size guide.

Data Source	Maximum SQL Connection
Object Server	30
DB2	12

Table 3 Data Source Connection Pool

For a cluster setup, the connection pool sizes are automatically replicated to secondary servers. Each Impact server in the cluster will make its own connections to the event database as set by the connection pool settings. For example, a maximum connection pool size of 12 with 2 servers in the cluster, there can be 24 connections made by Impact. The connection pool size can be adjusted by editing the data source configuration from the Netcool/Impact UI.

The threads for each Event Processor Service on each cluster member are managed individually. You must set each secondary server Event Processor Service minimum and maximum thread count manually. The Event Processor Service minimum and maximum threads are not automatically replicated to secondary servers. These settings must be added to the secondary event processor service properties file manually. This file is also located in the $IMPACT_HOME/etc directory. The file is called "<instance_name>_eventprocessor.props." where "instance_name" is the name of your Impact instance. The secondary server has to be restarted for this change to take effect. Alternatively, you can use the Command Line Service to connect to the secondary server and issue commands to change the minimum or maximum number of Event Processor threads and also can stop and restart the Event Processor service. When updated via the Command Line Service, there is no need to restart the secondary server. Just restarting the Event Processor service will make use of the configuration updates.

APPENDIX

Product Documentation and Impact Support Links

- **Tivoli Netcool/Impact Wiki**

  https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Tivoli%20Netcool%20Impact/page/Home

- **Tivoli Netcool/Impact Documents**

  http://www-01.ibm.com/support/knowledgecenter/SSSHYH/welcome?lang=en

- **Tivoli Netcool/Impact 7.1.0 Support**

  http://www-947.ibm.com/support/entry/portal/product/tivoli/tivoli_netcool~impact?productContext=-69484059

- **IBM Pattern Modeling and Analysis Tool for Java Garbage Collector**

  PMAT

Notices

Trademarks