IBM Business Analytics Proven Practices: IBM Cognos BI 10.2.1.x JMeter Stability Package

Product(s): IBM Cognos BI 10.2.1.x; Area of Interest: Performance

Business Analytics Proven Practices Team
Business Analytics Proven Practices Team
IBM

26 June 2014

This stability test designed for IBM Cognos 10.2.1.x demonstrates how to use standard JMeter to stability test an IBM Cognos BI system of that version. No knowledge prerequisites required.

View more content in this series

Introduction

Purpose of Document

This document accompanies a file representing an Apache JMeter test plan to probe an IBM Cognos BI Server install for stability under a given load. After providing some background on the tool and presenting enough knowledge about IBM Cognos BI report execution so that one can comprehend the scope of the test and its features, this document will provide details on how to use Apache JMeter to run the provided test plan and configure it for a given environment. Based on best practice recommendations regarding test execution, guidelines for result interpretation are provided along with troubleshooting tips.

The test plan and process described in this document has been used successfully in a number of client engagements and in instances of scalability testing led by IBM Cognos and is therefore considered a Best Practice.

This stability test plan is a good start whenever the answer to any one of the following questions is “yes”:

- There is no other load and stability testing software such as Rational Performance Tester or Mercury LoadRunner available.
- The system exhibits unstable behavior under load, requests fail, response times are slow, unexpected error messages occur and/or one needs a mechanism to force the errors to occur in order to troubleshoot.
• The IBM Cognos BI install needs to be checked for stability and load before moving it to the next release stage such as Testing or Pre-Production.
• One wants to test scalability and needs quick answers on the performance impact of configuration or architectural changes (such as adding another instance of Report Service).

Applicability

The test plan must be run by JMeter version 2.7 or above but JMeter 2.11 is recommended. The plan is not compatible with earlier versions of JMeter.

JMeter is available on almost every platform as it's a Java based tool and therefore the plan can be run from all supported JMeter platforms as well.

The test plan has been recently tested on various operating systems and configurations and is designed for IBM Cognos BI 10.2.1.x (Refresh Pack 1 including Fixpacks 1, 2 and 3) only. It is suitable for testing regardless of set-up topology, mix of operating systems or bitness in use. The use of the Cognos Application Firewall (CAF) is fully supported as it is best practice and strongly suggested to leave CAF enabled all times.

Exclusions and Exceptions

The test plan presented as part of this package was designed for the specified IBM Cognos BI server releases. Running it against other versions of IBM Cognos BI won't necessarily work due to more or less subtle changes to the requests, their payload syntax and their specifics which may change from version to version. Use the plan for versions explicitly listed in this document only. Previous versions of this stability package exist for older versions of IBM Cognos BI.

The test plan currently runs reports from the sample packages provided with the product in IBM Cognos Viewer. This includes

• Compatible Query Mode (CQM) reports
• Dynamic Query Mode (DQM) reports
• Dynamic Cube (DYNC) reports
• Cognos Workspaces (refer to plan details)

It does not support the following

• Running any of the Studios
• Active Reports
• SSL enabled Gateways
• Single Sign-on for authentication

The test plan is provided “as-is” and its sole purpose is to provide a quick, versatile and flexible, free and easy means for checking stability of a possibly complex install of IBM Cognos BI Server and it’s stability in load situations. There is no formal support of this plan. If you are interested in detailed, individual load-testing and performance tuning exercises, please contact your IBM Cognos representative.
The test plan is considered to be a starting point for looking into stability and system operation in complex environments. Any more serious analysis, in particular in regards to performance, should be part of an IBM Services engagement and should be done using a commercial load testing tool such as IBM Rational Performance Tester. While the results from running the provided test plan may provide valuable indications about IBM Cognos BI configuration issues, necessary third-party configuration/tuning and possible stability issues, you should always bring those indications forward to IBM in order to interpret the indications and to resolve any issues. IBM Cognos BI is a complex product and results or issues that may surface by running the test plan may not always relate to a single specific reason and it often requires extensive knowledge of the IBM Cognos BI architecture and process flow in order to interpret the results correctly. Searching the IBM Information Center and the IBM Cognos Support web site for error messages encountered is always encouraged.

Assumptions
It's assumed that the reader is familiar with the concepts described in the IBM Cognos BI Architecture and Security Guide, in particular the IBM Cognos BI components and services.

Background
This section provides some background information about the JMeter tool and the IBM Cognos BI report execution process, both of which are required to comprehend the implications running this plan and its results may uncover.

Apache JMeter
JMeter is a Java based tool developed in an open source software project of the Apache organization. Its purpose is to issue requests to a server and collect and record the response. JMeter supports HTML, XML, SOAP, JDBC and other request types. It can execute multiple parallel threads and therefore simulate multiple clients hitting a system. It allows building test plans manually or by recording requests of a single client session through some proxy. Plans can be enriched with programmatic logic like conditional execution, response parsing using regular expressions and much more.

The home of the JMeter tool can be found in the Resources section at the end of this document. JMeter is being actively developed, meaning new versions become available from time to time. While new versions claim to be backward compatible, it is strongly advised to use the most recent version to benefit from fixes and performance improvements as well as added functionality. For details about using JMeter and what each of the elements in a plan is doing, please consult the JMeter documentation.

JMeter uses a concept known as a Thread Group to simulate a set of clients which run through a certain sequence of steps. A single client is represented by a single thread which sequentially iterates through defined steps. A Thread Group maintains a configurable amount of these client threads. A single test plan can define multiple Thread Groups which then could be run on different JMeter installations on separate hosts. However, it is currently not possible to fork multiple threads from a step of a plan, functionality that would be required to, for example, simulate AJAX type requests in HTML pages.
IBM Cognos BI Report Execution

IBM Cognos BI uses the concept of **conversations** when executing reports of any type. A conversation is an operation consisting of multiple steps, each step technically a single request. Thus executing a report becomes a series of requests and responses rather than a single request with a single response.

Conversations start with a **primary request** and can have one to many **secondary requests** belonging to the same conversation. Each conversation starts out in **synchronous** mode, which means that the sender (the client) of the request waits for the recipient (the addressed service) to respond. During that time the client is blocked. Since this is inefficient if kept up for more than a few seconds, there is a configurable time-out known as the **Primary Wait Threshold** (PWT) after which a conversation will go into **asynchronous** mode. At that time the sequence of requests changes.

When the conversation has gone asynchronous, the client will have to send **secondary requests** at specific intervals to keep the conversation active. If those secondary requests cease, the receiving service will stop processing the current request and the conversation will end with an error instead of a result. By default, the secondary request has a time-out of 30 seconds. This timeout is known as the **Secondary Wait Threshold** (SWT). Every 30 seconds the client needs to send a **wait** request to the same service to keep the conversation alive and indicate it is still waiting for the result. The target service continues to process and signals when it completed processing the current request and the result can be retrieved by the client. How many iterations of wait requests will be required entirely depends on the system resources. If system load is high, e.g. many concurrent users, the system may require more time to process a single conversation than it would if there was only one user using the system. That is, each conversation will potentially look different at the HTML level on every run depending on the actual system load. This is the main reason why one cannot simply record an IBM Cognos BI report execution with a tool like JMeter or Rational Performance Tester and play it back for a test. If a primary request is not answered within the time limit of the PWT, the conversation will have to go asynchronous, however if this is not recorded, the request will fail. Each conversation can be different in the number of required requests each time it gets executed.

All of this is reflected by the requests which make up a conversation. If the client is a browser running the IBM Cognos Viewer tool, the handling of the conversation is done by JavaScript embedded in the HTML pages that make up the IBM Cognos Viewer tool. If the client is an IBM Cognos SDK program, the conversation handling has to be implemented by the client itself. This applies to the JMeter plan as well. At this point the JMeter plan has to mimic the JavaScript functionality by carrying forth specific HTML form variables and dynamically react to their contents along with server responses.

Conversations are used whenever something is **executed** - for example reports, agents, analyses and workspaces. When IBM Cognos services interact amongst each other, conversations happen as well and each of the services implements the client side of conversation handling.
The details of processing a conversations in a client application is covered in the *IBM Cognos SDK Developer Guide* which is part of the IBM Cognos SDK product. However, for the purposes of this document, the above information is sufficient.

**Preparing the environment**

**Test environment setup considerations**

Running a JMeter test has some implications regarding the tool itself and the setup of IBM Cognos BI.

**Where to install JMeter**

During the test the JMeter application will simulate concurrent clients connecting to IBM Cognos BI using a HTTP client. JMeter spawns a thread for each simulated client session (concurrent user) so sufficient resources regarding CPU and RAM will be necessary to avoid bottlenecks on the JMeter side. Platforms which provide fast threading support, such as Linux, will usually perform better for this task and greater amounts of RAM will further positively impact the performance of JMeter due to context switches being served from memory. As a rule of thumb, CPU speeds of 1.5 GHz and beyond as well as RAM sizes of 2GB at least are suitable for running JMeter in a scenario simulating about 30 clients - the actual requirements depend on the number of clients which shall be simulated though. These are recommendations based on experience rather than through formal, wide-scale testing.

To get the most out of the test, the JMeter application should be run from a separate machine where no IBM Cognos BI components or databases are installed. As stated above, depending on the number of users you want to simulate, there could be a large number of threads and sockets being used which may significantly impact the machine performance and if the box is also busy running IBM Cognos BI components, adding the additional resource demand of JMeter is not a good idea.

**IBM Cognos BI Gateway implementation**

When simulating many clients who access IBM Cognos BI through a gateway, there is one thing to be aware of. If using a CGI gateway the web server will spawn one instance of the CGI executable for each client session - each instance will be a separate process with its own memory and resources. It’s therefore advised to follow the best practice of leveraging non-CGI IBM Cognos BI Gateway implementations (MOD, MOD2 or MOD2_2 for Apache web servers or ISAPI for Microsoft IIS) as they provide much better resource usage and are specifically coded for performance. CGI gateways are generic and supported by every web server but they are not suitable for production systems and even less for systems exposed to heavy load. The test plan will work with any IBM Cognos BI Gateway as well as directly against an IBM Cognos Dispatcher but simulating many (> 30) concurrent client sessions may overload your entry point.

**Use monitoring tools**

While testing with the provided plan, a closer watch to the resources consumed at the operating system level and at the query and/or Content Manager database is advised. Imposing a higher
load onto the system may identify bottlenecks or challenges you haven't seen before. You should have sufficient access and privileges to run administration tools such as IBM DB2 Control Center, IBM Data Studio, Oracle Enterprise Manager, Microsoft SQL Enterprise Console and various operating system monitoring commands.

Launch a browser on a separate machine and use the IBM Cognos Administration tool to monitor things such as,

- number of Report Service processes spawned
- numbers of threads opened for each Report Service process
- number of threads opened for the IBM Cognos BI Services
- number of connections to the query and content store DB
- memory consumption on the Content Manager and Dispatchers
- number of interactive tasks
- memory consumption of Query Service process
- Dynamic Cubes cache hit/miss ratio

This is not a comprehensive list but it provides a reasonable start. For more detailed monitoring of IBM Cognos BI using Java Monitoring Extensions (JMX), please refer to the System Management Methodology link provided in the Resources section.

**Implementing the Prerequisites**

**Install JMeter**

JMeter is a Java based tool and you need to have a working Java Runtime Environment (JRE) to run JMeter. JMeter 2.7 onwards requires at least JRE 1.6. After JMeter is installed, you make need to configure it for use of the proper JRE. More details about JMeter prerequisites and a head start on installing and using it can be found in the JMeter User Manual (see Resources section).

The development of this version of the plan was done using JMeter 2.9 and 2.11 using IBM JRE 1.6.0 64-bit on Red Hat Enterprise Linux and using IBM JRE 1.7.0.55 64-bit on Microsoft Windows 2008-64.

**Install the required IBM Cognos BI sample packages**

The test plan provided with this document runs reports from the following sample deployments provided with IBM Cognos BI.

- IBM_Cognos_Samples
  - Go Data Warehouse (query), GO Data Warehouse (analysis) and Go Sales Query packages
- IBM_Cognos_Samples_DQ
  - GO Data Warehouse (analysis) and Go Sales Query packages
- IBM_Cognos_Power_Cube
  - Sales and Marketing (cube) package
- IBM_Cognos_DynamicCube
• Go Data Warehouse Sales package

If a certain type of report is not to be tested, the samples of the corresponding sample deployment may be omitted.

While the GO Data Warehouse packages and the GO Sales package use relational data sources that may require some additional setup, the Sales and Marketing package is based on an IBM Cognos PowerCube and no additional setup is required.

Installation instructions for the relational and PowerCube sample packages can be found in Appendix D of the IBM Cognos 10.2.1 Installation and Configuration Guide or in the online Information Center for IBM Cognos 10.2.1 at, http://www.ibm.com/support/knowledgecenter/api/content/SSEP7J_10.2.1/com.ibm.swg.ba.cognos.inst_cr_winux.10.2.1.doc/c_settingupsamplesbi.html#SettingUpSamples

Installation instructions for setting up the dynamic cube samples can be found in Chapter 3 of the Dynamic Query Guide or in the online Information Center for IBM Cognos 10.2.1 at http://www.ibm.com/support/knowledgecenter/SSEP7J_10.2.1/com.ibm.swg.ba.cognos.dyn_query.10.2.1.doc/c_dqm_samples.html

Configuration of test plan

After you started JMeter and loaded the test plan you'll get presented with the JMeter user interface. As shown in Illustration 1, the left pane allows you to browse the test plan in a hierarchical tree structure. In the right pane the properties and settings of the element selected in the left pane will be presented.
Illustration 1: JMeter expanded tree of test plan elements

JMeter expands all the elements in the tree pane by default as of version 2.1.2. This default behavior makes getting an overall view of the test plan more difficult so it is recommended to tell JMeter to not expand all elements on startup. To do this, start JMeter with the command line switch `-Jonload.expandtree=false` or edit the file `jmeter.properties` and add the line `onload.expandtree=false`.

Once JMeter opens up, expand the top level node **IBM Cognos 10.2-1-x – BI Stability package** which will display another node of the same name. Expand this as well. Now, expand the **Overall Tests** element and you'll have a good overview of the plan as shown in Illustration 2.
Once all elements of the second level have been collapsed this will provide a good overview of the test plan's structure. There are four general configuration elements, one top element containing all the test plan steps and two elements involved with visualizing the test results.

In Illustration 2, the main configuration element **Test configuration Parameters** (highlighted in red) is involved when adjusting the plan to a specific environment by editing it's properties (highlighted in yellow). More detail on these configuration properties will be provided in the Configure Parameters section. The test itself features two elements which handle logon and logoff requests (highlighted green) which will be explained in the Configure Parameters section and fifteen elements that execute a report (highlighted in light blue) which will be discussed in the next section.

**Understand the plan structure and disable specific tests**

The test plan runs fifteen different reports using different output formats. Some of the reports include prompting. Following is the list of reports run along with some details about each report execution regarding output format and the steps performed. Be aware that charts and any binary output such as Excel, CSV or PDF are rendered by the Report Server and streamed to the client as a binary file in response to an explicit GET request. In a browser this would be achieved by iframe(s) or JavaScript issuing those GET requests to a particular service which provides those items from internal caches in which they have been created.

Reports using Dynamic Query Mode (DQM) start at Element 10, Reports against Dynamic Cubes (DYNC) start at element 20 and Workspace reports start at element 30.
1. Budget vs Actual, GO Data Warehouse (analysis)
   • Is run in HTML format.
   • Does not involve prompting.
   • One Next Page operation, once the first page is served.
   • A very simple cross-tab report.

2. Order Invoices - Donald Chow, Sales Person, GO Sales (query)
   • Is run in PDF format.
   • Does not involve prompting.
   • The PDF (257 pages) is retrieved from the Content Store.
   • A basic list report that is fairly large with some layout and branding.

3. Planned Headcount, GO Data Warehouse (analysis)
   • Is run in HTML format.
   • Does not involve prompting.
   • A report containing 5 charts will get rendered. The plan will download the individual GIF representations of the charts to simulate the viewing of the HTML report output.

4. Global Bonus Report, Go Data Warehouse (analysis)
   • Is run in HTML format.
   • Has one prompt page with two parameters, Year and Region. Values used are 2005 for Year and Americas for Region.
   • A list report.

5. Pension Plan, GO Data Warehouse (query)
   • Is run in CSV format. No prompting. A list report with plenty of data.

6. Recruitment Report, GO Data Warehouse (analysis)
   • Is run in XLWA format.
   • Has one prompt page with one parameter, Year. The value used for Year is 2006.
   • A report containing several graphs and a table.

7. Historical Revenue, Sales and Marketing (cube)
   • Is run in HTML format.
   • Has two prompt pages with one parameter each. The first prompt page takes the year to run report for and the second prompt page is for providing the month of the year selected on the first prompt page. Values used for the year is 2006 and the month is June.
   • A single chart report where the chart gets retrieved explicitly by a separate GET request.

8. Customer Returns and Satisfaction, GO Data Warehouse (analysis)
   • This is a Dynamic Query Mode report.
   • Is run in HTML format.
   • Has no prompt page but a Next Page operation is done once the first page of results is available.
   • A report containing 2 charts and two lists, the charts get retrieved explicitly by a separate GET request.

9. Return Quantity by Order Method, GO Data Warehouse (analysis)
   • This is a Dynamic Query Mode report.
   • Is run in XLWA format.
   • Contains a single cross-tab.

10. Employee Training by Year, GO Data Warehouse (analysis)
    • This is a Dynamic Query Mode report.
• Is run in HTML format.
• Has two prompt pages, the first page prompts for a year and the second page prompts for a quarter of that year. Values being used for the year is 2012 and the quarter is 3.
• A report containing a charts and a cross tab. The chart is retrieved with a separate GET request.

11. Order Invoices – Donald Chow, GO Sales (query)
• This is a Dynamic Query Mode report.
• Is run in PDF format.
• Has no prompt page.
• A report containing a larger list and since it’s a PDF, all result data must be loaded.

12. Revenue by order method and region, GO Data Warehouse Sales
• Uses a Dynamic Cube.
• Is run in PDF format.
• Has no prompt page.
• A small report containing a single cross tab. Ideal for scaling up.

13. Revenue by retailer and product line, GO Data Warehouse Sales
• Uses a Dynamic Cube.
• Is run in HTML format.
• Has no prompt page but a Next Page operation is done once the first page of results is available.
• A report containing a larger list.

14. Sales by Year Workspace, Cognos Workspace Samples folder
• This is a Workspace report.
• Contains 9 Widgets of which 6 produce charts.
• Widgets get executed sequentially, charts get retrieved explicitly with a separate GET request.

15. Marketing Workspace, Cognos Workspace Samples folder
• This is a Workspace report.
• Contains 6 Widgets, of which 3 produce charts plus one cross tab.
• Widgets get executed sequentially, charts and cross tab get retrieved explicitly with a separate GET request.

The test plan is displayed in a hierarchical tree structure. The top element is Overall Test which contains all the sub-elements. The sub-elements are numbered and elements 00 and 99 are used for authentication only (refer to the Configure Authentication section for details). All other elements execute a specific report or Workspace. If you want to exclude a report from the test, this is easily achieved by right-clicking on the element and selecting Disable from the context menu (refer to Illustration 3). Those reports will be skipped when running the test.
Illustration 3: JMeter's right-click context menu showing the Disable and Toggle options for the selected element

You will see the disabled element being greyed out in the left tree pane. You can re-enable this element by accessing the same context-menu again by right clicking on it and selecting Enable. Recent versions of JMeter introduced the Toggle action which has a convenient shortcut (CTRL + T) to toggle an element between enabled and disabled.

Since the test plan can be perceived as a hierarchical tree, one element or node can contain other nodes which become child elements in that context. The part of the test plan which executes reports is a single element labeled Overall Tests of type transaction controller. This basically means that all its child elements get treated as one single transaction in regards to measuring execution time and outcome. This top element has one numbered sub-element per report executed during the test - these per report sub-elements are all of type transaction controller. This allows one to get an overall time total for all steps of the conversation executing the specific report regardless of report details and steps. As an example, consider a plan that has been configured to only run sub-elements 01 and 02. In this case, the transaction controllers would provide a time total for the execution of all steps required to execute Report_1, a time total for the execution of all steps required to execute Report_2 and an overall time total for the whole test.
At the very end of the test plan you will find two **Listener** elements. These elements will record every request, the response and present the data in some user friendly form. The **View Result Tree** element will present a tree-view of the plan's requests being issued and the **Aggregate Report** element will show a view similar to a list report. The section titled **Result Interpretation** has more details about these Listener elements. Additional Listener elements can be added by right-clicking and adding them from the Listener category. Choices include graphs and statistical views. Refer to the JMeter documentation for further details.

**Configure Parameters**

The element labelled **Test configuration Parameters** is one of the two places that will need to be edited to configure the plan for a given environment. There are typically five parameters which need to be specified and tailored to suit a given environment.

- **Servername**
  Specify the NetBIOS or fully qualified server name for your entry point to IBM Cognos BI. This can be any Gateway or an external Dispatcher URI.
- **Port**
  The network port for the entry point.
- **url**
  The URL to the entry point. This usually is everything that follows the server name and includes a leading “/” character. For an IBM Cognos BI Gateway, the form is `/<alias>/cgi-bin/<gateway_executable>` (for example `/ibmcognos/cgi-bin/mod2_2_cognos.so`). For an IBM Cognos BI Dispatcher, the form is `/<context_root>/servlet/dispatch/ext` (for example `/p2pd/servlet/dispatch/ext`).
- **Namespace**
  This is the ID property of the namespace used for authentication, if any, for the IBM Cognos BI product. If there are multiple namespaces configured, only one can be chosen as authentication to multiple namespaces is not supported by the plan. The namespace ID can be obtained from IBM Cognos Configuration. If you want to run without authentication leave this blank. Refer to the **Configure Authentication** section for further details.
- **Templocation**
  As of IBM Cognos BI version 10.1, temporary objects for the report execution are now stored on the local file system instead of the Content Store. This means that by default, charts, graphs and other binary outputs which are part of the report get stored in the Cognos temporary folder configured locally using IBM Cognos Configuration instead of storing these in the Content Store as was the case in IBM Cognos 8 BI. This affects the URLs used to retrieve those parts when displaying the report. The test plan has to use the setting which matches the configuration of IBM Cognos 10 BI to be able to retrieve the charts and binary outputs. If this property is not set correctly, there won't be any samples labeled **Retrieve graph output** in the result tree viewer. While this doesn't make the test plan fail, it might skew the results or prevent the detection of Graphics Service failures to render the chart since missing charts won't be anticipated. Always set Templocation to a value of **FS** unless IBM Cognos 10 BI has been configured to use the Content Store for these temporary objects, in which case Templocation should be set to a value of **CM**. Illustration 4 shows the value of the **Temporary objects location** property in the IBM Cognos Administration tool.
Illustration 4: IBM Cognos Administration showing for the Temporary object location being set to the default value of Server File System

There are additional parameters which are used for some more advanced features. You normally don't need to change these parameters but they can be adjusted if you understand the impact of the change.

- **Outputlocale**
  With this setting you can specify the output locale to use for the report outputs, overriding the output locale specified in the profile of the IBM Cognos 10 BI user account used by the test plan to logon.

- **pwt**
  This specifies the Primary Wait Threshold (PWT) for the report execution conversation. IBM Cognos Viewer uses a value of 3 seconds, other clients usually use 7 seconds. If set to 0 (zero) this will prevent the conversation from going asynchronous and thus block resources in the system and having each user thread wait for the request result. This will lead to less performance but is sometimes used to troubleshoot issues.

- **swt**
  This specifies the Secondary Wait Threshold (SWT) for the report execution conversation. The default is 30 seconds and should not be changed. Lowering the value does not help to retrieve results any faster or earlier as any service can signal to the waiting client that it has completed working on the request independent from the SWT. Increasing the value does not necessarily save resources by less frequent reloads of heartbeat requests being sent. This property is there primarily for troubleshooting purposes.

**Configure Authentication**

If the test does not involve authentication - that is IBM Cognos 10 BI allows for anonymous access - the plan's namespace parameter must be left blank in the **Test configuration Parameters** element. The test plan will react accordingly and skip the authentication step which automatically makes the session to be authenticated as anonymous.

If the test does involve authentication, one must provide the namespace ID of the namespace to authenticate against in the plan's namespace parameter in the **Test configuration Parameters** element as described earlier in the **Configure Parameters** section.
In addition, one will need to provide user names and passwords to the plan to be used for establishing authenticated sessions to IBM Cognos BI. Please note that Single Sign-on to IBM Cognos BI is not supported by the plan and therefore explicit credentials such as username and password are required.

When the plan is executed, every thread will simulate a single user's session. To authenticate that session a valid IBM Cognos BI user login is required. Since IBM Cognos BI supports a single user to have multiple active concurrent sessions at a time, theoretically the whole test plan can be run using a single user account. Best practice is however to not have a single user account to be used more than 5 times concurrently. Ideally to improve troubleshooting and test quality regarding auditing the results and having a resemblance to a true multi-user use case, each client session simulated by JMeter should use a separate set of user credentials. A big reason for using separate credentials is to mitigate against the effects of user specific caching that may occur internally in IBM Cognos BI.

There are two possibilities for providing user credentials to the plan. The choice depends on the number of user threads to be simulated during the test and the number of user credential sets available to be used.

For only a few sets of user credentials, one can conveniently use a User Parameters element in JMeter. This element allows to specify a defined set of variables per user in a simple table. An initial column labeled Name defines the set of parameters, one per row, every additional column defines a separate user.

For this plan, one must define two variables, user and pass, to hold the user name and the corresponding password. As an example, a test with 5 sets of user credentials would require this element to define a table with 2 rows and 5 columns, excluding the initial Name column. By clicking on the Add User button, one can add additional columns which get labeled automatically by JMeter. This is demonstrated in Illustration 5. User_1 has a user attribute whose value is dooku and a pass attribute that has a value of count.

Illustration 5: The User Parameters element of the plan containing a table defining a single user's credentials

If more than a few user credential sets are to be used, it is better to specify usernames and passwords in a separate CSV file. For this, one must create a simple text file, which should be
saved into the same folder as the test plan itself. Into the CSV file, one adds the usernames and passwords, one set of credentials per line which has to be terminated by a CR/LF pair. In JMeter, deactivate the User Parameters element by right-clicking on it and selecting Disable and then enable the CSV Data Set Config element labeled User Credentials File by right-clicking on it and selecting Enable. As shown in Illustration 6, in the CSV Data Set Config element, specify the name of the CSV file containing the user credentials in the Filename property. If the CSV file was saved in the same folder as the test plan file, specifying the file name only is sufficient as relative file names are resolved based on the plan's file system path. For the Variable Names property, one must specify user,pass which means that every line read from the file will be expected to contain two strings separated by a comma. The first string will be used for username, the second for password. It is important to know that that the passwords will be stored in clear text and there is no support for encrypting this file so it will be necessary to secure the CSV file properly using file system security.

Illustration 6: The CSV Data Set Config element labeled User Credentials File showing the element's properties

The CSV file used in this example, 1021xusers.txt, would contain something like this:

user1,password1
user2,password2
...
user99,password99

Keep in mind that spaces are not ignored with the default settings, so accidental spaces behind the separating comma may lead to the login failing. If commas or spaces are required in the strings, put them in quotes and enable the Allow quoted data? property in the CSV Data Set Config element by switching it to True.

During plan execution, each thread will read a single line from the file. By using the Recycle on EOF option, one can either make the file roll over upon reaching its end or make the plan stop reading values from the CSV file. It's highly recommended to enable the Recycle on EOF option to prevent authentication failures due to missing credentials.
Regardless of which method is used to provide user credentials to the plan, ensure that all credentials have been verified to work before starting the plan in order to save time troubleshooting.

**Configure number of threads and ramp up**

By default, the test plan runs a single thread and runs through all enabled plan elements once. It's most likely that multiple users will need to be simulated so the number of threads and possibly the number of iterations will need to be increased. This can be done by clicking on the second **IBM Cognos 10.2.1.x - BI Stability package** element in the tree. This element is the **Thread Group** element, an element type that was discussed at the beginning of this document.

**Illustration 7: The properties of the Thread Group element including number of threads, ramp-up period and loop count**

![Thread Group properties](image)

The properties of the Thread Group element will be displayed on the right JMeter pane and the number of threads, ramp-up period and a loop count can be specified (Illustration 7). JMeter will distribute the creation of new threads evenly over the specified ramp-up time. So if there are 5 threads and a ramp-up period of 60 seconds has been defined, a new thread will be started every 12 seconds until the configured number of threads has been reached.

In addition to these parameters, there is the **Action to be taken after a Sampler error** setting which specifies what action to take if a thread hits an error. This is set to **Stop Thread** and you must not change it as it allows you to easily identify errors which occurred during the execution of the test. The plan has been built to verify successful execution of a request. If the result is something unexpected, the thread will fail and you can identify the exact position in the plan where this happened.

The possible maximum number of threads depends on the available resources of the system hosting JMeter. Of course, even if JMeter may be able to start hundreds of threads, this does not mean the IBM Cognos BI system targeted by the test is able to handle them, so the best practice is to increase the number of threads in steps with every run of the test plan. The ramp-up period will heavily influence the level of concurrency, lowering this period may easily overload both the JMeter machine and the target system so again an incremental approach is recommended.
A good practice when using/developing a plan is to start with a single user and one iteration. If this works fine, reach for more, try 10 users and 60 seconds ramp up. At the very least a single server BI system should very well be able to authenticate all sessions - the rest depends on the reports being run.

**Running the test**

The test plan can be started and stopped by using the **Run** menu.

All activity (ie: the result of each single request) will be recorded by the two Listener elements in the plan. It's a good practice to clear the recorded data before each test run. This can be done from the JMeter Run menu by selecting **Clear All**. The keyboard shortcuts are **CTRL+E** to clear the test and **CTRL+S** to start the test. The test can be stopped at any time by employing the **Stop** command from the Run menu or pressing **CTRL+** (period). If there are a high number of threads running it may take a while for the test to stop.

Upon start there will be a light green indicator in the upper right next to two numbers indicating how many threads of the maximum configured are actually running.

**Illustration 8: JMeter Status bar showing the activity indicator and thread counters to the farmost right**

As shown in Illustration 8, the numbers 11/30 mean 11 threads are running and there will be a maximum of 30 threads. If the number of threads running drops below the maximum number of threads after the initial ramp up period passes, you should check whether a thread completed or stopped due to an error. This can be done in the Listener elements.

Usually after starting the test you click on one of the Listener elements to monitor progress. While the **Aggregate Report** element provides more an overview, the **View Result Tree** renders a tree view of all the requests being issued and the responses received for them. This tree is updated on the fly so it grows while the test is running.

One important point to note is that an element will appear in the result tree only after the request has been sent and 1) a response has been received or 2) the request failed. For example, if you don't see the initial Login element appearing, it may be due to the fact that you are try to connect to a wrong URL and the web server time-out is 30 seconds.

One can click on each element in that result tree and the right pane will show information about the Sampler result (technical information such as headers and bytes transmitted), the request sent (the exact request URL and information transmitted) and the response data received on individual tabs. This is shown in Illustration 9. Investigation of the response data will help to determine why a certain element might have failed or what data has been returned. JMeter can render the response data to different formats to increase readability as well as providing the ability to search in the response data using regular expressions.
Several elements won’t have a response because they don’t state actual requests but just structure test plan requests. One example are the transaction controller type elements. Although they appear in the plan, they won’t have a result in terms of an HTTP request and response. They only structure the plan and calculate time totals.

If a request in the result tree listener is shown in red, this indicates that the request was not successful and the thread which issued the request was stopped. Looking at the response data tab of that request will yield information about what went wrong.

You may notice that the elements appear to be out of sequence when being listed in the result tree Listener. This is due to the fact that when the hierarchical tree structure of the plan is processed child elements are run first. Only after all child elements of an element have been processed, the parent is considered. Therefore a parent element will show up after all its child elements. Illustration 9 depicts this way of reading the result tree.

Illustration 9: The result tree Listener output showing responses in an indented list of numbered elements

By using indentation and numbering the plan was designed to help reading the result tree more easily. Every element has a number, every child element has been assigned the same number and
an additional second level number, separated by a dot. In addition, child elements get indented to the right by several spaces per level in the hierarchy. For example, x.1 means the first sub element of element x, it displays indented by 3 spaces to the right. This scheme applies to all elements and up to 5 levels of indentation and numbering.

Consider the Logon element with the number 00 (zero zero) of the plan. This element has two child elements 0.1 and 0.2. The first child element is nested in a condition which means it will only get executed if the condition is true. In the result tree view this Logon element appears as either one or two entries for the sub-elements, depending on whether anonymous authentication is used or not, in which case 0.1 won't get executed. In the result tree view the actual Logon element will only appear after those child elements if the Logon actually succeeds. When reading top to bottom the occurrence of the 00 element therefore implies that the complete Logon element was processed successfully. If at any point the authentication would have failed, that parent 00 element for Logon would be missing and one of the child elements would be printed in red with a warning symbol in front of it. Summary elements such as 00 - Logon are displayed unindented and by anticipating the indentation the blocks or groups formed for each plan element become clear. Illustration 10 shows red rectangles around the blocks for elements 00 to 03.

The Aggregate Report Listener has a table-like output, showing a single row per distinct test element with the columns containing several timing values such as minimum, maximum, median and average (mean) times. These values give you timing information related to the overall execution of a report as well as detailed information for specific elements or groups of elements. This is shown in Illustration 10. In addition, one can also learn how often an element was executed and determine if all configured threads have passed by a certain element yet.
Finally, one more clarification regarding the output provided by the Listeners. When running tests using only a single thread (user), the order of the entries appearing in either the Aggregate Result Listener or the View Result Tree Listener will reflect the structure of the test plan in the way that was described earlier. If a test is using multiple threads the sequence of the entries in either listener may appear to be mixed up. This is not an issue with the plan, it is caused by the many threads running concurrently. They will file their result updates to the Listener independently and therefore an entry with a higher number may appear before one with a lower number simply because every thread is executing a different element of the plan. This is expected behavior and applies in particular to the elements which execute the Asynchronous Conversation. Those elements may appear later in the execution of the plan because when more load is introduced to the system, requests which used to complete more or less instantly now may take several seconds to complete. There is a way to label the element outputs by their thread number but that would have cluttered the plan and the author decided against it. If required, one can prefix each element label with \[\{\$\_threadNum\}\] to add the number of the executing thread of a thread group to the output.

### Result Interpretation

After the test has completed the best place to start gathering result information is the Aggregate Result Listener. It will hold information about how long each request, each operation and the overall test took to run as well as statistical aggregates such as median and average. These timing metrics will be the main indicators of performance.
Since we are aiming at stability here, the error column should show 0% entries only. If there are errors for a particular row, the error column will show something different than 0%. Switch to the View Result Tree Listener and look for red entries. Investigate the responses and react to the error messages visible there by adjusting IBM Cognos 10 BI configuration or system parameters.

To get additional value from the results, you could compare the timing between different setups, such as changing the Gateway type or adding another Report Server. This will demonstrate how performance is influenced. One should always start with a baseline test and set of results. One simple way to establish a baseline is to perform three runs of the same test and noting the results. Each set of results should be similar. After that, one can start changing the system or configuration to observe the impact, positive or negative, of the changes.

Stability wise the test should always complete without any errors. If specific reports fail repeatedly, all but the failing elements should be disabled to narrow down the issue.

If a test shows some errors it does not necessarily mean that the targeted IBM Cognos BI system is unstable. It simply indicates that further investigation is due. Most errors can be remedied by adjusting the IBM Cognos BI configuration as some of them may be the result of overall system bottlenecks. In all cases consult the section titled Troubleshooting and the IBM Cognos related Technotes before taking action. Search for the error codes read from the response data tab in the result tree of failing requests. If unsure, log a PMR with IBM Cognos Support.

In cases where BiBusTkServerMain processes crash and leave a minidump (Windows) or a core (UNIX/Linux), you should contact IBM Cognos Support immediately to log a PMR. You will need to provide the test plan along with comprehensive information about the system configuration.

In summary, the results of the test will provide an understanding of the stability of a given IBM Cognos BI system under certain circumstances. It has to be emphasized that this test plan is just a sample and can never substitute for an on-site engagement from trained IBM Cognos staff or partners to run complex performance and scalability tests. Reach out to your IBM Cognos representative to arrange for an on-site engagement if you need solid and confirmed statements regarding stability and performance.

Appendix A - Troubleshooting

This section contains some of the most common error messages and hints to solve the cause of the error. It is by no means a comprehensive guide, just a quick reference. For details look up any errors returned by IBM Cognos BI in Technotes or contact IBM Cognos Support on them. Again, this plan itself is NOT supported.

JMeter errors

- JMeter user interface is not fully functional - some menu options such as Open are grayed out. The console shows a Java exception.
  Action: This is an issue with the start script (jmeter.bat/.sh) which must be fixed in the script. If the machine that JMeter is running on has more than one JRE, it's possible that the wrong JRE is taking precedence from the PATH. This was the case for the author and it was fixed by...
putting the command `set JM_LAUNCH=%JAVA_HOME%\bin\%JM_LAUNCH%` into the script before it calls the JRE and set JAVA_HOME at the beginning of the script.

- JMeter doesn't start.
  Action: Ensure you're using JRE version 1.6. or higher.
- JMeter UI is slow to respond or doesn't respond at all.
  Action: Consider running JMeter from a more powerful computer or limit the number of threads (users) to between 20 and 30.
- In the Result Tree Listener, test plan elements lack requests for fetching graphs or binary outputs.
  Action: Verify that the `templocation` setting in the plan matches the IBM Cognos 10 configuration for the Temporary object location property.

IBM Cognos errors, seen as error pages in the result data tab of a request when displayed as HTML:

- 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.
  Action: Increase the Dispatcher Queue time-out.
- Error: RSV-ERR-0021 "The server cannot find a primary request with ID <someID>.")
  Action: Verify that your application server thread limit is not exceeded.

Plan errors, items fail indicated by red color in the listener results:

- The first thing to do is test whether the report actually works using a browser and the same login credentials. Only if this works flawlessly 3 times in a row using a new browser session each time, one should assume this is not caused by the actual report in the BI system.
- If the report works using a browser, disable all other elements but the one containing the failing item. This will help to narrow down the root cause.
- If the element fails repeatedly and it contains prompts, use Report Studio or Query Studio to double check the prompt values passed by the plan. Find the data item used for the prompt filter. Next identify the Member Unique Name (MUN) for the desired prompt value and paste it into the plan. As an alternative, try running the report via URL passing the prompt values as parameters on the URL. If that fails, fix the URL and paste the correct MUN into the plan.
## Downloads

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cognos BI 10.2.1 JMeter Stability package</td>
<td>Cognos_10.2.1-BI_Stability.zip</td>
<td>66KB</td>
</tr>
</tbody>
</table>
Resources

- Apache JMeter Project
- Apache JMeter Documentation
- IBM Cognos BI 10.2.1 Architecture and Security Guide
- IBM Cognos BI System Management Methodology
About the author

Business Analytics Proven Practices Team

© Copyright IBM Corporation 2014
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)