WebSphere Application Server for z/OS

WebSphere Liberty Batch: Common Problems

This document can be found on the Web at:
www.ibm.com/support/techdocs
Search for document number **WP102760** under the category of "White Papers"

Version Date: November 12, 2018
See Document change history on page 8 for a description of the changes in this version of the document

IBM Software Group
Application and Integration Middleware Software

David Follis
IBM Poughkeepsie
845-435-5462
follis@us.ibm.com

Don Bagwell
IBM Washington System Center
dbagwell@us.ibm.com

Many thanks go to ...Scott Kurz, Andy Mauer, Dan Belina, Chris Gianfrancesco, and all the customers who had problems so we would know which ones were 'common'….

Quick Reference

BBOA1REG RC:12, RSN:10	Incorrect WOLA 3-part name
BBOA1CNG RC:12, RSN:14	Missing WOLA CBIND authorization
HTTP 401 error	Incorrect security info for batchManager
HTTP 401 + CWWKY0304W	Missing batch role authorization
HTTP 500 + Failed to load JPA	Missing Job Repository tables
Dispatcher/Executor hung CLI	Properties don't match message selector Batch Events not configured correctly
Bad job message delivered endlessly	Configure a backout threshold and queue

Version Date: 11/12/2018

Introduction

We decided we wanted to create a document that cataloged some problems people had with WebSphere Liberty Batch, their symptoms, and how to resolve the problems. As we collected things to include, it turned into a messy, unorganized pile. To try to organize it, we decided to create a story. Here we go..

Our hero (or heroine, as you like) has a JSR-352 Java Batch application that has been shown to work quite well in a development environment. A pre-production test environment has been created. Nobody would push an application straight to production without testing it first, at least not in the fantasy land of our story. Unfortunately, when trying to get the job to run, there are problems. It can't be the application because it worked just fine in development. It must be something wrong with the environment (and odds are it probably really is). What could it be?

The rest of this paper is kind of like one of those choose-your-own-ending stories. You find the type of problem and, hopefully, the specific symptom you have, and we will, hopefully, guide you to a happy ending to the story.

Good Luck!

You Can't Get There From Here – Network Issues

You are probably using the batchManager or batchManagerZos Command Line Interface (CLI) to submit the job. To get that done, the CLI will need to contact the target server, whether that is the server that will run the job or a dispatcher that will queue the job to run in another server. Just making this connection is a place where things can go wrong.

batchManagerZos Connectivity Issues

If you are using batchManagerZos to interact with a Liberty server, the most common problems relate to WOLA. WOLA, or WebSphere Optimized Local Adapter, is the cross-memory communications mechanism that batchManagerZos uses to talk to the target server. To make a connection to the server, the batchManagerZos uses the BBOA1REG API to register with WOLA and then the BBOA1CNG API to get a connection. Problems using WOLA will most likely surface as bad return/reason codes from those services.

The most basic error is that the server is not properly set up to use WOLA or else the CLI doesn't have the right name to connect to it. A server is configured with a 3-part name that identifies it to clients connecting via WOLA. Either of these problems looks the same to a client: no server with the specified 3-part name can be found. It will look like this in the

Version Date: 11/12/2018

output from batchManagerZos (where our 3-part name is LIBERTY+BATCH+MANAGER and you can see below we've got a typo and it is XIBERTY instead).

```
ERROR: [jnu_wolaRegister(WolaName *, _CHAR12 *)] rc=12  BBOA1REG RC:12,
RSN:10, registrationName:13b6926b batchManager, wolaName:[XIBERTY ]
[BATCH   ] [MANAGER ]
```

The other likely problem is security related. To be allowed to connect to a server using WOLA, you need permission to the right profile in the CBIND class. The profile name will be BBG.WOLA.THREE.PART.NAME where those last three qualifiers match the three part WOLA name of the server you are connecting to.

If you don't have access, you'll get an error like this:

```
ERROR: [jnu_wolaGetConnection(_CHAR12 *, int, _CHAR12 *)] rc=12  BBOA1CNG
RC:12, RSN:14, registrationName:13b6926b batchManager, waitTime:30,
connHandle:13b6927
```

The WOLA return and reason codes are pretty good about differentiating between different error situations. If you get one that isn't included here, look up the service name and the return/reason code you got.

Who Do You Think You Are? – Authorization Issues

The userid running the CLI needs to have access to the server being contacted. And that same user needs to have permission to submit jobs. Let's go through the symptoms for a few common problems..

In this case, the userid provided to the batchManager CLI is not known to the server or the password provided is incorrect. Use a valid userid and password.

```
Error: Server returned HTTP response code: 401 for URL:
https://localhost:10443/ibm/api/batch/jobinstances
java.io.IOException: Server returned HTTP response code: 401 for URL:
https://localhost:10443/ibm/api/batch/jobinstances
```

If you have a good userid and password, but the id is not authorized to submit jobs, you'll get the error below. Correct this by granting access to an appropriate batch role (batchAdmin or batchSubmitter). In a development environment you might not have

needed to configure any roles if application security was not enabled. Note that application security can sometimes be enabled indirectly through the activation of other features.

```
Error: Server returned HTTP response code: 401 for URL:
https://localhost:10443/ibm/api/batch/jobinstances: [Error 401:
CWWKY0304W: User Bob is not authorized to start batch jobs.]
```

Brains! Brains! – Job Repository Issues

The Job Repository is where Liberty Batch keeps track of the status of jobs it is currently running as well as jobs run in the past. In a development environment batch applications can be run using an in-memory repository that requires no set up at all. In a production environment you will want to use a repository located in a database. For that to work, the tables have to be set up properly.

To be sure you are not using the in-memory Job Repository, verify that the following message appears in the messages.log for the server:

```
CWWKY0005I: The batch JPA persistence service is activated.
```

If a persistent Job Repository is being used, but the tables are not defined (and the configuration doesn't allow them to be dynamically created), you'll get the error below trying to submit a job. This is the same error you get if you have created the tables, but the server configuration isn't actually pointing to them because something is set incorrectly, such as the schema name. This error will come out for whatever the first attempt is to access the Repository, whether it is to submit a job, list jobs, etc.

```
Error: Server returned HTTP response code: 500 for URL:
https://localhost:10443/ibm/api/batch/jobinstances: [Error 500:
javax.batch.operations.BatchRuntimeException: Failed to load JPA
PersistenceServiceUnit
```

It's All About The Execution – Dispatcher/Executor Model Issues

In development a single server is usually used to run the job. But in production multiple servers might be involved, usually in a dispatcher/executor model. In this configuration the CLI contacts the dispatcher to submit the job, then the dispatcher puts a message representing the job on a queue, and finally an executor server picks up the message and dispatches the job. There is a lot that has to be set up properly for that to work.

Unfortunately, the most common symptom of something being set up incorrectly is that the CLI will hang waiting for the job to finish. The trick then is to figure out where things got stuck. The most obvious is that the message representing the job got put on a queue and never picked up by an executor server. Use MQExplorer or a similar tool to look at the queue (if you can) and see if there is a message stuck there. Examine the properties on

Version Date: 11/12/2018

the message and make sure they match what is specified in the message selector in the Activation Specification in the executor server.

If you are using batchManagerZos and waiting for a job-completion batch event to notify the CLI that the job is done and the job finished but the CLI is still waiting, there's something wrong with the event configuration. Verify the dispatcher and executor server are successfully publishing batch events. Verify that the queue manager and topic root being used by the CLI is the same as where the Liberty servers are publishing batch events. Security around the messaging engine might be a factor also.

Too Much of a Bad Thing

There might be cases where the message representing a job gets delivered to a server but encounters an error trying to get the job running. Some of these errors will just immediately cause the job to fail. An example might be that the application name you specified (and matched in the message selector if you are picking up jobs using the application name) is wrong and the server can't find that application. The job fails and the message is considered 'processed'.

But other errors might cause the batch container to just throw an exception back to the messaging engine leaving the message undelivered. In this case the message will likely just be re-delivered to the server. If whatever is wrong happens again (and again and again) then this process will just go on and on.

To stop the loop, look at the configuration for the job message queue. Configure a backout threshold for the queue. That indicates how many tries to make delivering a message that fails trying to be processed. Also configure a backout requeue queue. This is the queue where these troublesome messages will be deposited for later examination.

Conclusion

And the job ran successfully in pre-production test, the production environment was created exactly like the test environment, the job moved into production without a hitch, and they all lived happily ever after.

Document change history

Check the date in the footer of the document for the version of the document.

July 3, 2018	Original Version
July 9, 2018	Updated document number
November 12, 2018	Added backout requeue threshold info

End of WP102760