# High Availability with Liberty

| Updated January 8, 2016 by dmazzella | Tags: *None*

Page Actions

## Background:

This wiki entry is provided as-is with no warranty or support. It is intended to be used by IT Professionals with years of experience in J2EE application server configuration, database configuration, web server configuration. This document is ***not*** intended to be a "basic" or "easy" to use guide, it is an advanced, cliff-notes style entry to point out tips and tricks to get a Liberty collective (elastic cluster) going. High Availability is a complex configuration to setup, and it will vary from environment to environment. This is not designed to be a cookie cutter step by step documentation, but rather a guide to show it is possible to setup and configure TRIRIGA with a HA Application Server and Web Server layer.

Setup and configuration of a TRIRIGA High Availability Cluster is not an easy task, and requires deep knowledge of session replication, how application servers communicate to other servers, and how IBM TRIRIGA Application Platform works in general. Please visit the reference materials at the end of this article, and try things in a test/dev environment so you have a good understanding of how things works. It is estimated that it could take a week or two of work to setup and understand this the first time. Please pace yourself, and re-read this wiki and other links multiple times.

Also know that this setup isn't used to get better performance, but high availability. In some cases, High Availability setups put a higher stress on the overall hardware and network, causing slower performance than without the HA setup in place.

Here are the general steps used to setup a proof of concept TRIRIGA/Liberty cluster setup.

**What you need:**

1 or more servers to act as the Web Server - Install IHS 8.5.5 and make sure to update to the latest 8.5.5.8 fix pack.

1 or more servers to act as the Application Servers - Install TRIRIGA 3.5.0 or better on each server.

2 database servers, one for the TRIRIGA Schema, one for the SESSION database - It is best practice to have the SESSION replication database on its own database server.

## High level steps:

1. Update the first WebSphere Liberty installation to provide the features for High Availability by adding a collective controller. Lets assume the TRIRIGA Installation was done prior to this, the installation was made choosing Liberty as the application server and DB2 as the database, and /tririga as the context-path. No modifications have been done to the install at this point. We are also assuming you have installed into the **~/ibm/tririga1/** directory on Linux.

a. Cd into the TRIRIGA Install, under the Liberty bin directory. You will now install the features adminCenter-1.0, collectiveController-1.0, and clusterMember-1.0. Run these commands:

```
cd ~/ibm/tririga1/wlp/bin
./featureManager install --acceptLicense adminCenter-1.0
./featureManager install --acceptLicense collectiveController-1.0
./featureManager install --acceptLicense clusterMember-1.0
./featureManager install --acceptLicense dynamicRouting-1.0
```

These command will connect to the Liberty repository and install the features into the bundled version of Liberty.

b. Create a controller server, this is a separate Liberty instance to act as the controller for the Collective and cluster. Run these two commands to create the controller

```
cd ~/ibm/tririga1/wlp/bin
./server create controller1
./collective create controller1 --keystorePassword=~s3cret~ --createConfigFile
```

This will create the cluster controller in the ~/ibm/tririga1/wlp/usr/servers/controller1 directory

c. Edit **~/ibm/tririga1/wlp/usr/servers/controller1/server.xml** to enable the collective. Add the following lines after the 2nd line of the file.

```
vim ~/ibm/tririga1/wlp/usr/servers/controller1/server.xml
```

Note: You can edit with vim or emacs, you don't need to use gedit if you know those other editors better.

On the line before the very last line **</server>** add these lines.

```
<include location="${server.config.dir}/collective-create-include.xml" />
<quickStartSecurity userName="system" userPassword="~s3cret~" />
<featureManager>
  <feature>ssl-1.0</feature>
  <feature>adminCenter-1.0</feature>
  <feature>dynamicRouting-1.0</feature>
</featureManager>
```

d. Start the controller via the command

```
~/ibm/tririga1/wlp/bin/server start controller1
```

These commands setup and started the controller used to handle the clustering commands and routing.

2. Now, we have to update the tririgaServer instance to replicate the sessions via a database, and join the collective. We will add the features that we added above, and setup the Session replication database. Liberty should create the table in the session database automatically.

a. Edit **~/ibm/tririga1/wlp/usr/servers/tririgaServer/server.xml** to use a its own session replication database:

compare_revisions
customizations customize
database db2 exchange
find_available_times gantt_chart
gantt_scheduler group
memory_footprint modifications
modify object_label
object_revision
operating_system oracle
performance platform
problem_determination reports
reserve reserve_performance
revision revisioning
single_sign-on snapshot space
sql_server sso support system
system_performance
tags: track_customizations
tririga troubleshoot tuning
upgrade ux version versioning
Cloud | List

Members

```
vim ~/ibm/tririga1/wlp/usr/servers/tririgaServer/server.xml
```

On the line before the very last line **</server>** add these lines

```
<featureManager>
<feature>ssl-1.0</feature>
<feature>sessionDatabase-1.0</feature>
<feature>collectiveMember-1.0</feature>
<feature>clusterMember-1.0</feature>
<feature>adminCenter-1.0</feature>
</featureManager>
    <dataSource id="SessionDS" jndiName="SessionDS"
          transactional="true"
          type="javax.sql.DataSource"
        statementCacheSize="10"
        isolationLevel="TRANSACTION_READ_COMMITTED"
        commitOrRollbackOnCleanup="commit"
        syncQueryTimeoutWithTransactionTimeout="true"
        supplementalJDBCTrace="false">
    <jdbcDriver id="db2" libraryRef="tririgaLib" />
    <properties.db2.jcc  databaseName="session" serverName="localhost" portNumber="50505" user="session" password="~s3cret~" />
    <connectionManager maxPoolSize="100"
              minPoolSize="10"
              purgePolicy="ValidateAllConnections"
              connectionTimeout="10s"
                          maxIdleTime="30m"
              agedTimeout="30m"/>
    </dataSource>
  <httpSessionDatabase id="SessionDB" dataSourceRef="SessionDS" scheduleInvalidation="true" />
```

It is best practice to use a different physical database server for the session database, since there could be a very active connection.

b. **vim ~/ibm/tririga1/wlp/usr/servers/tririgaServer/bootstrap.properties**

Add thses two lines, setting the http port number and unique cloneId:

```
httpPort=8001
cloneId=tririgaServer1
```

c. **vim ~/ibm/tririga1/config/TRIRIGAWEB.properties**

To make sure the agent manager works property in this VM, update the TRIRIGAWEB.properties setting the INSTANCE_ID and NAME to a unique name

Find these two lines and update them as such:

```
INSTANCE_ID=10001
INSTANCE_NAME=tririgaServer1
```

3. To join the tririgaServer 1 to the collective, first make sure the collective controller (controller1) is started, then run the command:

```
cd ~/ibm/tririga1/wlp/bin

export JVM_ARGS=-Dcom.ibm.websphere.\
collective.utility.autoAcceptCertificates=true

./server status controller1

./collective join tririgaServer --host=localhost --port=9443 \
  --user=system  \
  --password=~s3cret~ --keystorePassword=~s3cret~ --createConfigFile
```

4. Add the result returned in the previous step, and the following lines to the tririgaServer's server.xml

```
vim ~/ibm/tririga1/wlp/usr/servers/tririgaServer/server.xml
```

On the line before the very last line </server> add these lines

```
<!-- CLUSTER SETUP -->
<featureManager>
  <feature>clusterMember-1.0</feature>
</featureManager>

<quickStartSecurity userName="system" userPassword="~s3cret~" />
<keyStore id="defaultKeyStore" password="~s3cret~" />
<include location="${server.config.dir}\collective-join-include.xml" />
```

5. Start the Liberty tririgaServer

```
cd ~/ibm/tririga1/wlp/bin
./run.sh
```

6. Configure IHS to point to dynamically route to the different Liberty collective members.

Run the dynamicRouting setup command on one of the controllers to generate the keystore and plug-in configuration files.:

```
cd ~/ibm/tririga1/wlp/bin
./dynamicRouting setup --port=9443 --host=localhost \
  --user=system       --password=~s3cret~ \
  --keystorePassword=~s3cret~ \
  --pluginInstallRoot=/opt/ibm/WebSphere/Plugins \
      --webServerNames=webserver1

cp plugin*  /tmp
chmod a+r /tmp/plugin*
```

7. Double click on the "root console" icon on the desktop. Copy the generated plugin-key.jks and plugin-cfg.xml files from /tmp to the web server plugin directory.

a. In the root console, cd to the web server bin dir, make sure the seLinux Boolean is set to allow httpd to connect out, then run gskcmd to convert the keystore to CMS format and to set personal certificate as the default. CMS format is the supported format of the WebSphere Plug-in.

```
cd /opt/ibm/HTTPServer/bin

setsebool -P httpd_can_network_connect on

./gskcmd -keydb -convert -pw ~s3cret~ -db /tmp/plugin-key.jks\
 -old_format jks -target /tmp/plugin-key.kdb -new_format cms\
 -stash

./gskcmd -cert -setdefault -pw ~s3cret~ -db /tmp/plugin-key.kdb \
-label default
```

b. Copy the plugin-key.kdb, plugin-key.rdb, and plugin-key.sth files that are created by gskcmd from the /tmp directory to the directory /opt/ibm/pluginInstallRootargument>/config/<web server name>/

```
cp  /tmp/plugin-key.* /opt/ibm/WebSphere/Plugins/config/webserver1/
```

Press y each time to overwrite the files.

c. Copy the /tmp/plugin-cfg.xml to the directory specified in the WebSpherePluginConfig directive in the IBM HTTP Server (IHS) httpd.conf file. The plugin-cfg.xml is generated with the <IntelligentManagement> stanza. When Dynamic Routing is enabled in a collective, there is one <Connector> stanza for each collective controller.

```
cp /tmp/plugin-cfg.xml /opt/ibm/WebSphere/Plugins/config/webserver1
```

d. Start the web server and begin routing to the application installed in the collective.

```
cd /opt/ibm/HTTPServer/bin/
./apachectl restart
```

8. At this point you can open Firefox, and go to http://localhost/tririga
The Liberty controller and IHS will work to deliver the application from the tririgaServer to Firefox.

9. Next, we will add the *second* tririgaServer to the collective. We assume you have installed the second tririga into the **~/ibm/tririga2** directory.

   a. Step 1, tell this Liberty instance to join the collective

   ```
   cd ~/ibm/tririga2/wlp/bin

   export JVM_ARGS=-Dcom.ibm.websphere.\
   collective.utility.autoAcceptCertificates=true

   ./collective join tririgaServer --host=localhost --port=9443 \
     --user=system  \
     --password=~s3cret~ --keystorePassword=~s3cret~ --createConfigFile
   ```

   b. We have to setup all the clustering and session fail over work. We will do this in this one step for the server.xml. We have to add the result returned in the previous step, and the following lines to the tririgaServer's server.xml,

   ```
   vim ~/ibm/tririga2/wlp/usr/servers/tririgaServer/server.xml
   ```

   On the line before the very last line </server> add these lines

   ```
   <featureManager>
   <feature>ssl-1.0</feature>
   <feature>sessionDatabase-1.0</feature>
   <feature>collectiveMember-1.0</feature>
   <feature>clusterMember-1.0</feature>
   <feature>adminCenter-1.0</feature>
   </featureManager>

       <dataSource id="SessionDS" jndiName="SessionDS"

             transactional="true"

             type="javax.sql.DataSource"

           statementCacheSize="10"

           isolationLevel="TRANSACTION_READ_COMMITTED"

           commitOrRollbackOnCleanup="commit"

           syncQueryTimeoutWithTransactionTimeout="true"

           supplementalJDBCTrace="false">

       <jdbcDriver id="db2" libraryRef="tririgaLib" />

       <properties.db2.jcc  databaseName="session" serverName="localhost" portNumber="50505" user="session" password="~s3cret~" />

       <connectionManager maxPoolSize="100"

                 minPoolSize="10"

                 purgePolicy="ValidateAllConnections"

                 connectionTimeout="10s"

                             maxIdleTime="30m"

                 agedTimeout="30m"/>

       </dataSource>

     <httpSessionDatabase id="SessionDB" dataSourceRef="SessionDS" scheduleInvalidation="true" />

   <!-- CLUSTER SETUP -->
   <featureManager>
     <feature>clusterMember-1.0</feature>
   </featureManager>

   <quickStartSecurity userName="system" userPassword="~s3cret~" />
   <keyStore id="defaultKeyStore" password="~s3cret~" />
   <include location="${server.config.dir}\collective-join-include.xml" />
   ```

   c. **vim ~/ibm/tririga2/wlp/usr/servers/tririgaServer/bootstrap.properties**

   Add thses two lines, setting the http port number and unique cloneId:

   ```
   httpPort=9001
   cloneId=tririgaServer2
   ```

   d. **vim ~/ibm/tririga2/config/TRIRIGAWEB.properties**

   To make sure the agent manager works property in this VM, update the TRIRIGAWEB.properties setting the INSTANCE_ID and NAME to a unique name

   Find these two lines and update them as such:

   ```
   INSTANCE_ID=10002
   INSTANCE_NAME=tririgaServer2
   ```

   e. start the tririga2 Liberty service

   ```
   cd ~/ibm/tririga2/wlp/bin
   ./run.sh
   ```

11. You now have dynamically added the second TRIRIGA Liberty instance to the dynamic collective. Lets explore what this configuration brings you.

   a. In Firefox, Click on the Liberty Admin Center bookmark, or type in https://localhost:9443/adminCenter
   b. Login as **system** with the password of **~s3cret~**
   c. Click on the Explore link. Here you should see 1 Application running, 1 Cluster, 3 servers running, and 1 host.
   d. Click on 1 Applications link. Here you should see the ibm-tririga running on 2 instances, on the defaultCluster
   e. Click the back button, and click on Clusters, then click on defaultCluster. Here you should see 1 Server, and 1 Application.
   f. Click on the Servers icon on the left, it should show tririgaServer, Running ibm-tririga on the defaultCluster on localhost.
   g. Click back until you are back at the main Explore page, then click on Servers. Here you should see three servers, controller1, and two tririgaServers. Clocl on tririga1
   h. On the left, click on the Monitor Icon. Here you can monitor Heap, Loaded Classes, Active Threads, and CPU usage.

12. We will now show the failover by shutting down one of the tririgaServer instances.

   a. Open the a console
   b. Find the PID of the tririgaServers by running the command

   ```
   ps a | grep tririga | grep java | grep -v controller1 | awk '{print $1}'
   ```

   c. The results of this command should show to PIDs of Liberty
   d. Open Firefox, and click on the (IHS) TRIRIGA Admin Console bookmark, or go to http://localhost/tririga/html/en/default/admin
   e. Login with system/admin
   f. On the right Look at the value of Java Process ID on Host, and see that it matches one of the results from step b.
   g. Back in the "root console" run the command

   ```
   kill -9 #####
   ```

   Where **#####** is the PID number you see from step f.
   This will kill that tririgaServer instance.

   h. Back in Firefox, refresh the page, or click on Admin Summary again
   i. Notice that you did not have to login again. Also notice the Java Process ID changed to the other tririgaServer.

In conclusion

One the IHS, and liberty collective controller are configured, additional tririgaServer instances can be brought up without having to restart either the collective controller or IHS. This enables you to have a dynamic collective. Additional HA tuning can be done to have multiple collective controllers, multiple IHS servers, and real network HA equipment to provide true failover.

Reference Materials

WASDev.net - http://wasdev.net/

IBM HTTP Server http://www-01.ibm.com/support/knowledgecenter/SSEQTJ_8.5.5/as_ditamaps/was855_welcome_ihs.html

Installing IBM HTTP Server http://www-01.ibm.com/support/knowledgecenter/SSEQTJ_8.5.5/com.ibm.websphere.ihs.doc/ihs/welc6miginstallihsdist.html

Configuring IHS and the WebSphere plugin for use with Liberty http://www14.software.ibm.com/webapp/wsbroker/redirect?version=phil&product=was-base-dist&topic=twlp_admin_webserver_plugin

Dynamic Routing configuration http://www-01.ibm.com/support/knowledgecenter/was_beta_liberty/com.ibm.websphere.wlp.nd.multiplatform.doc/ae/twlp_wve_enabledynrout.html

[VIDEO] Enabling IHS for Liberty Dynamic Routing - https://www.youtube.com/watch?v=v6CenrZcuYA

**wongkw** commented on April 6, 2016 Permalink

In this example, both the INSTANCE_ID are above 10000. In the TRIRIGAWEB.properties documentation (https://www.ibm.com/support/knowledgecenter/SSHEB3_3.5.0/com.ibm.tap.doc/ins_configure/r_cfg_tririgaweb_properties.html), it is stated that <quote>The INSTANCE_ID property must be set to a numeric value less than 10000.</quote>

Which is correct?