Application Level Messaging

Each Yellowfin node must have the **ClusterManagement** servlet enabled for the node to become “Cluster Aware”. The ClusterManagement servlet is enabled by adding an additional configuration block to the web.xml file on each node.

Application messaging is performed differently depending on the implementation mode. There are currently three modes available — REPOSITORY, DYNAMIC and LEGACY.

Multicast cluster messaging (DYNAMIC mode)

Yellowfin application messaging is handled by a multicast messaging library called JGroups. Using this method will automatically find other nodes in the cluster sharing the same Yellowfin database.

The default configuration of JGroups uses UDP multicast messages to determine group membership and find new nodes. There may be environments where these types of messages cannot be sent. For example, Amazon does not allow multicast packets on its internal network between nodes. The Multicast Cluster Messaging adapter allows you to pass an XML configuration file to configure JGroups to use other methods for node discovery. This file can be referenced by passing the path to the BroadcastConfiguration servlet parameter within the ClusterManagement servlet on each node’s web.xml file.

The following servlet definition needs to be added to the web.xml file **on each node**:

```
<!-- Cluster Management -->
<servlet>
       <servlet-name>ClusterManagement</servlet-name>
       <servlet-class>com.hof.mi.servlet.ClusterManagement</servlet-class>
       <init-param>
             <param-name>ClusterType</param-name>
             <param-value>DYNAMIC</param-value>
       </init-param>
       <init-param>
             <param-name>SerialiseWebserviceSessions</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>CheckSumRows</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>EncryptSessionId</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>EncryptSessionData</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>AutoTaskDelegation</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
              <param-name>TaskTypes</param-name>
              <param-value>
                     REPORT_BROADCAST_BROADCASTTASK,
                     REPORT_BROADCAST_MIREPORTTASK,
                     FILTER_CACHE,
                     SOURCE_FILTER_REFRESH,
                     SOURCE_FILTER_UPDATE_REMINDER,
                     THIRD_PARTY_AUTORUN,
                     ORGREF_CODE_REFRESH,
                     ETL_PROCESS_TASK,
                     SIGNALS_DCR_TASK,
                     SIGNALS_ANALYSIS_TASK,
                     SIGNALS_CLEANUP_TASK,
                     COMPOSITE_VIEW_REFRESH
                           </param-value>
       </init-param>
       <init-param>
              <param-name>MaxParallelTaskCounts</param-name>
              <param-value>
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2
              </param-value>
       </init-param>
       <load-on-startup>11</load-on-startup>
 </servlet>
```

TaskTypes and **MaxParallelTaskCounts** must be fine-tuned for each node, based on which tasks, the deployment of Yellowfin is expected to run. The configuration specified above allows the node to run a maximum of two tasks of each type, in parallel. For detailed information, see here.

Multicast with repository discovery (REPOSITORY mode)

Repository discovery is an implementation of DYNAMIC mode, but with a custom plugin for discovering nodes via the shared Yellowfin repository. This can be useful for enabling clustering on environments where multicast packets do not work.

This functionality can also be enabled with DYNAMIC mode with the RepositoryDiscovery servlet parameter set to true on each node's web.xml file.

The following servlet definition needs to be added to the web.xml **on each node**:

```
<!-- Cluster Management -->
<servlet>
       <servlet-name>ClusterManagement</servlet-name>
       <servlet-class>com.hof.mi.servlet.ClusterManagement</servlet-class>
       <init-param>
             <param-name>ClusterType</param-name>
             <param-value>REPOSITORY</param-value>
       </init-param>
       <init-param>
             <param-name>SerialiseWebserviceSessions</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>CheckSumRows</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>EncryptSessionId</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>EncryptSessionData</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
             <param-name>AutoTaskDelegation</param-name>
             <param-value>true</param-value>
       </init-param>
       <init-param>
              <param-name>TaskTypes</param-name>
              <param-value>
                     REPORT_BROADCAST_BROADCASTTASK,
                     REPORT_BROADCAST_MIREPORTTASK,
                     FILTER_CACHE,
                     SOURCE_FILTER_REFRESH,
                     SOURCE_FILTER_UPDATE_REMINDER,
                     THIRD_PARTY_AUTORUN,
                     ORGREF_CODE_REFRESH,
                     ETL_PROCESS_TASK,
                     SIGNALS_DCR_TASK,
                     SIGNALS_ANALYSIS_TASK,
                     SIGNALS_CLEANUP_TASK,
                     COMPOSITE_VIEW_REFRESH
                            </param-value>
       </init-param>
       <init-param>
              <param-name>MaxParallelTaskCounts</param-name>
              <param-value>
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2,
                     2
              </param-value>
       </init-param>
       <load-on-startup>11</load-on-startup>
 </servlet>
```

TaskTypes and **MaxParallelTaskCounts** must be fine-tuned for each node, based on which tasks, the deployment of Yellowfin is expected to run. The configuration specified above allows the node to run a maximum of two tasks of each type, in parallel. For detailed information, see here.

Web service cluster messaging (LEGACY mode)

Yellowfin's legacy cluster messaging is handled by AXIS web services. This requires that all nodes be defined at start-up, and that the service end-point, port, user and password be defined in each node's web.xml file. Legacy mode does not allow cluster instances to reside on the same host.

The following servlet definition needs to be added to the web.xml **on each node**:

```
<!-- Cluster Management -->
<servlet>
	<servlet-name>ClusterManagement</servlet-name>
	<servlet-class>com.hof.mi.servlet.ClusterManagement</servlet-class>
	<init-param>
		<param-name>ServiceUser</param-name>
		<param-value>admin@yellowfin.com.au</param-value>
	</init-param>
	<init-param>
		<param-name>ServicePassword</param-name>
		<param-value>test</param-value>
	</init-param>
	<init-param>
		<param-name>ServiceAddress</param-name>
		<param-value>/services/AdministrationService</param-value>
	</init-param>
	<init-param>
		<param-name>ServicePort</param-name>
		<param-value>80</param-value>
	</init-param>
	<init-param>
		<param-name>ClusterHosts</param-name>
		<param-value>
			192.168.4.184
		</param-value>
	</init-param>
	<init-param>
		<param-name>SerialiseWebserviceSessions</param-name>
		<param-value>true</param-value>
	</init-param>
	<init-param>
		<param-name>CheckSumRows</param-name>
		<param-value>true</param-value>
	</init-param>
	<init-param>
		<param-name>EncryptSessionId</param-name>
		<param-value>true</param-value>
	</init-param>
	<init-param>
		<param-name>EncryptSessionData</param-name>
		<param-value>true</param-value>
	</init-param>
	<load-on-startup>11</load-on-startup>
 </servlet>
```

Parameters within web.xml

The following properties outline the options in the ClusterManagement servlet definition that can be set on each node's web.xml file.

Property	Value
ClusterType	DYNAMIC, REPOSITORY, or LEGACY. DYNAMIC will use multicast messaging and automatically find other nodes in the cluster. REPOSITORY is an extension of DYNAMIC where multicast messaging doesn't work. LEGACY is the default, and will use web services to communicate with a defined list of cluster nodes.
BroadcastConfiguration	A JGroups configuration file. This allows for a custom Jgroups configuration to be used in environments where multicast networking is not available. This is for DYNAMIC mode only and is optional. By default, JGroups will use the configuration defined in udp.xml.
ServiceUser	User that will validate the web service connection to other nodes. For LEGACY mode only.
ServicePassword	Password for the ServiceUser. For LEGACY mode only.
ServicePasswordEncrypted	True/False. Is the contents of the ServicePassword encrypted.
ServiceAddress	Address of the Yellowfin web service. For LEGACY mode only.
ServicePort	Port on which Yellowfin is running. For LEGACY mode only.
ClusterHosts	This is a comma-separated list of all nodes in the cluster. These can include IP addresses or hostnames. For LEGACY mode only.
SerialiseWebserviceSessions	True/False. This is required if using single sign-on on the cluster. It can serialise tokens to the database so that the token can be accessed from any node.
CheckSumRows	True/False. Security option to check sum the serialised web service session records in the database. This helps prevent modification to the table which could lead to the creation of unauthorized sessions in Yellowfin.
EncryptSessionId	True/False. Security option to encrypt the serialised web service session id in the database. This helps prevent modification to the table which could lead to the creation of unauthorized sessions in Yellowfin.
EncryptSessionData	True/False. Security option to encrypt the serialised web service session records in the database. This helps prevent modification to the table which could lead to the creation of unauthorized sessions in Yellowfin.
AutoTaskDelegation	True/False. This is only for DYNAMIC and REPOSITORY mode and is optional. When enabled, the cluster will automatically assign a “master” node which coordinates background task execution. When this is turned on, you do not need to manually configure a node to run background tasks. The master node also runs system tasks, provided they have not been explicitly disabled. When AutoTaskDelegation is disabled, each node operates like a LEGACY node. A specific node has to be designated for background task execution. This mode is not recommended.
TaskTypes	This option is only for DYNAMIC and REPOSITORY modes. It is a comma-separated list of task types which can run on the node. If this is not specified, the node will not run any background tasks. Available types are specified here.
MaxParallelTaskCounts	This option is only for DYNAMIC and REPOSITORY modes. It is a comma-separated list of numbers, which indicates how many tasks of each type (specified in TaskTypes), can concurrently run on the node. Each value specified in this list has a one-to-one mapping to values in the TaskTypes parameter. TaskTypes which do not have a corresponding count, are ignored.
SessionReplication	True/False. Set this to true if container-level session replication is enabled. This will modify the logic used for destroying sessions in a cluster.

Repository Discovery	True/False This option is for DYNAMIC mode only. This enables RepositoryDiscovery on builds prior to the REPOSITORY mode being supported. REPOSITORY mode is an alias for DYNAMIC mode with RepositoryDiscovery enabled.

Additional parameters

DYNAMIC mode may not work in some environments. This will usually be due to the networking configuration on the server. By default, JGroups will use IPv6 if it is available. IPv4 can be forced, which may allow for it to work correctly. Add the following to the catalina.sh or catalina.bat files:

File	Command
Catalina.sh	`JAVA_OPTS="$JAVA_OPTS -Djava.net.preferIPv4Stack=true"`
Catalina.bat	`set JAVA_OPTS=%JAVA_OPTS% -Djava.net.preferIPv4Stack=true`

When using the default Jgroups configuration (udp.xml), the multicast address and port can also be configured. This may help in environments where cluster nodes are not discovering each other.

File	Command
Catalina.sh	`JAVA_OPTS="$JAVA_OPTS -Djgroups.udp.mcast_addr=228.0.0.5 -Djgroups.udp.mcast_port=47885"`
Catalina.bat	`set JAVA_OPTS=%JAVA_OPTS% -Djgroups.udp.mcast_addr=228.0.0.5 -Djgroups.udp.mcast_port=47885`

Previous topic: Cluster installation
Next topic: Background tasks