Container Level Session Replication

Container Level Session Replication allows Yellowfin to be deployed in a fail-safe environment where in the event of a node in the cluster failing, a currently connected Yellowfin user can continue their session on another node. This is achieved with Session Replication at the Application Server level, and is not a function of Yellowfin itself, and hence is not a supported component of Yellowfin. This configuration also requires a load-balancer to evenly distribute incoming requests to different cluster nodes, which will not be addressed in this guide.

Different Application Servers may have different ways of implementing this functionality. This example is shown using Tomcat, which is shipped with Yellowfin.

Setting up session replication consists of the following steps:

1. Modifications to web.xml
2. Modifications to server.xml
3. Modifications to ROOT.xml

More information regarding Tomcat Session Replication can be found at http://tomcat.apache.org

Session Replication Limitations

There are several caveats to using Session Replication with Yellowfin (or any Java web application). The two main downsides of Session Replication are processing overhead, and memory usage.

There is additional processing overhead in replicating each session to all nodes in the cluster. This happens after each HTTP request, and by default, only the changed session elements are copied to other nodes. This means all nodes will be doing additional work each time a user does something on any connected node.

The function of Session Replication is to replicate a user's memory footprint on the server, to all nodes in the cluster. This means the sessions for all users in the cluster will be stored on each node, taking up additional memory. Expanding the cluster with additional nodes is then restricted by the amount of memory that each node has.

It should be considered if this overhead is worth the added fail-safe functionality that it brings. If Session Replication is not used, and a node fails, a user's session will be destroyed, but they'll be able to log in to another node to continue their work.

Modifications to web.xml

The text "<distributable/>" needs to be added to the web.xml file on each node underneath the "<webapp>" line. For example:

```
<web-app>
      <distributable/>
      <!-- System Event and Debug classes -->
      <listener>
            <listener-class>com.hof.servlet.SysSessionListener</listener-class>
      </listener>
```

To enable session replication, add the following to the **ClusterManagement** servlet in the web.xml file.

```
<init-param>
          <param-name>SessionReplication</param-name>
          <param-value>true</param-value>
</init-param>
```

Modifications to server.xml

Changes need to be made to the server.xml file in the **Yellowfin/appserver/conf** directory.

Add the additional XML entry within the "<host>" XML block.

```
<Cluster className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>
```

The IP address “228.0.0.4” is a multicast address that session replication messages will be received on. This does not need to be modified, but a Linux /Unix server may need to register a valid multicast route to receive broadcasts on this address. This can be done by running the following command on the console:

```
sudo route add -net 224.0.0.0 netmask 224.0.0.0 dev eth0
```

This syntax may be different for different flavours of Linux or Unnis. This example uses “eth0” as the network device that will be used for this route.

Modifications to ROOT.xml

Remove the lines from the ROOT.xml in the **Yellowfin/appserver/conf/Catalina/localhost** directory:

```
<Manager className="org.apache.catalina.session.PersistentManager" debug="0" distributable="false"
saveOnRestart="false">
      <Store className="org.apache.catalina.session.FileStore" />
</Manager>
```

These lines were previously added to suppress session serialization to disk in Yellowfin.

Additional Information

The Session Replication functionality in Tomcat may change in different versions or subversions. If the process outlined here is not working, check the Tomcat website for updated documentation for the version of Tomcat that Yellowfin is running on.

Previous topic: Background tasks