How to build an Apache TomEE cluster

If you have an application or a system and it is running, you surely want it to be available. Depending on the scenario, you need it to be highly available! Or perhaps you just want to balance the loading on your server.

In any of those cases a good choice maybe is to build a cluster. Just to give you an overview and in the case this is a new word for you, cluster is a set of independent servers that communicate to each other thru a network in order to make a service or a system to be more available than if you used just one server.

Each server of the cluster is called a node. That are some different types of cluster.

If you are a Java developer and are familiar with Tomcat you will find TomEE quite easy to use. They have almost the same structure and are very likely to setup. The "plus" for TomEE is: it is Java EE (6) compatible for web profile.

What does it mean? Means that it implements all the Java EE 6 web API's related. So if you are a Web Developer and wanna take advantage of Java EE 6 on your project, the Apache TomEE could be a good choice.

So you have an application, want to run it in a TomEE and want to do it in a cluster? Easy! Let's do it step by step.

- Download and install the JDK and the Apache TomEE.
  - The versions used were: JDK 1.8.73 and Apache TomEE 1.7.3 Web Profile. The SO was a Mac OS X 10.11.3 (El Captain).
- After installing try to run your TomEE and see if it is working;
- Go to your TomEE Home folder and edit the following file:
  - /conf/server.xml
- Add those lines to the "Engine" node:

```
<Cluster
className="org.apache.catalina.ha.tcp.SimpleTcpCluster"
channelSendOptions="6">

<Manager
className="org.apache.catalina.ha.session.BackupManager"
expireSessionsOnShutdown="false"
notifyListenersOnReplication="true"
mapSendOptions="6" />

<Channel
className="org.apache.catalina.tribes.group.GroupChannel">
```

```
<Membership
className=”org.apache.catalina.tribes.membership.McastService”
address=”228.0.0.4″
port=”45564″
frequency=”500″
dropTime=”3000″ />

<Receiver
className=”org.apache.catalina.tribes.transport.nio.NioReceiver”
address=”auto”
port=”5000″
selectorTimeout=”100″
maxThreads=”6″ />

<Sender
className=”org.apache.catalina.tribes.transport.ReplicationTransmitter”>

<Transport
className=”org.apache.catalina.tribes.transport.nio.PooledParallelSender” />
</Sender>

<Interceptor
className=”org.apache.catalina.tribes.group.interceptors.TcpFailureDetector
” />

<Interceptor
className=”org.apache.catalina.tribes.group.interceptors.MessageDispatch1
5Interceptor” />

<Interceptor
className=”org.apache.catalina.tribes.group.interceptors.ThroughputIntercep
tor” />

</Channel>

<Valve className=”org.apache.catalina.ha.tcp.ReplicationValve”
filter=”.*\.gif|.*\.js|.*\.jpeg|.*\.jpg|.*\.png|.*\.htm|.*\.html|.*\.css|.*\.txt” />

<Deployer
className=”org.apache.catalina.ha.deploy.FarmWarDeployer”
tempDir=”/tmp/war-temp/”
deployDir=”/tmp/war-deploy/”
watchDir=”/tmp/war-listen/”
watchEnabled=”false” />

<ClusterListener
className=”org.apache.catalina.ha.session.ClusterSessionListener” />

</Cluster>
```

- Save and close your file, then restart your TomEE. If everything is ok, your log file should have something like this:

  org.apache.catalina.ha.tcp.SimpleTcpCluster startInternal
  Cluster is about to start
  org.apache.catalina.tribes.transport.ReceiverBase bind
  Receiver Server Socket bound to:/192.168.0.104:5000
  org.apache.catalina.tribes.membership.McastServiceImpl setupSocket
  Setting cluster mcast soTimeout to 500
  org.apache.catalina.tribes.membership.McastServiceImpl waitForMembers
  Sleeping for 1000 milliseconds to establish cluster membership, start level:4
  org.apache.catalina.tribes.membership.McastServiceImpl waitForMembers
  Done sleeping, membership established, start level:4
  org.apache.catalina.tribes.membership.McastServiceImpl waitForMembers
  Sleeping for 1000 milliseconds to establish cluster membership, start level:8
  org.apache.catalina.tribes.membership.McastServiceImpl waitForMembers
  Done sleeping, membership established, start level:8

- Great! Now you have the first node of your cluster up and running. Not enough, right? So do it to another server, could be another machine or even a VM. Do it to as many servers as you want according to your needs. Each time you add a new node, the log file of other nodes should contain something like this:

  org.apache.catalina.tribes.io.BufferPool getBufferPool
  Created a buffer pool with max size:104857600 bytes of type:org.apache.catalina.tribes.io.BufferPool15Impl
  org.apache.catalina.ha.tcp.SimpleTcpCluster memberAdded
  Replication member added:org.apache.catalina.tribes.membership.MemberImpl[tcp://{192, 168, 0, 107}:5000,{192, 168, 0, 107},5000, alive=1016, securePort=-1, UDP Port=-1, id={111 -38 59 -66 -68 -29 72 -69 -103 12 -121 -120 -13 -25 -90 17 }, payload={}, command={}, domain={},]

- Now you have your beautiful cluster fully happy and working! There is just one thing missing: your application deployed to it. Simple:
  - Edit the web.xml of your application;
  - Add this attribute: <distributable />;
  - Save the file, close it and deploy to your cluster.

Dynamic and Static Discovery cluster with Apache TomEE

Now let’s see quickly how we can set them up at the Apache TomEE. So we will do just a little zoom to see where the things are done.

Dynamic Discovery

At the “Engine” node you should put this code:

```
<Cluster
className=”org.apache.catalina.ha.tcp.SimpleTcpCluster”
channelSendOptions=”6″>

<Manager…

<Channel…

<Valve…

<Deployer…

<ClusterListener…

</Cluster>
```

Static Discovery

At the same “Engine” node you should put this:

```
<Cluster
className=”org.apache.catalina.ha.tcp.SimpleTcpCluster”channelSendOption
s=”6″>

<Channel className=”org.apache.catalina.tribes.group.GroupChannel”>

<Interceptor
className=”org.apache.catalina.tribes.group.interceptors.StaticMembershipIn
terceptor”>

<Member className=”org.apache.catalina.tribes.membership.StaticMember”
port=”4000″ host=”server1″ uniqueId=”{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1}” />
<Member className=”org.apache.catalina.tribes.membership.StaticMember”
port=”4000″ host=”server2″ uniqueId=”{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2}” />
<Member className=”org.apache.catalina.tribes.membership.StaticMember”
port=”4000″ host=”server3″ uniqueId=”{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3}” />

</Interceptor>

</Channel>
```

</Cluster>

So as you can see, with the Static Discovery you use the StaticMember at the Member node instead of McastService used at the Dynamic Discovery.